Títol: Iris, aplicació per a la gestió de competicions de rem

Volum:	1/1
Alumne:	Joan Marc Montesó Gilabert
Director/Ponent:	Xavier Vilajosana
Departament:	Arquitectura de Computadors
Data:	20 juny de 2010

DADES DEL PROJECTE

Títol del Projecte: Iris, aplicació per a la gestió de competicions de rem

Nom de l'estudiant: Joan Marc Montesó Gilabert
Titulació: Enginyeria en Informàtica
Crèdits: 37.5
Director/Ponent: Xavier Vilajosana
Departament: Arquitectura de Computadors

MEMBRES DEL TRIBUNAL *(nom i signatura)*

President:

Vocal:

Secretari:

QUALIFICACIÓ

Qualificació numèrica:
Qualificació descriptiva:

Data:

ÍNDEX

1.- INTRODUCCIÓ

1.1.- Definició del projecte

El projecte Iris és un sistema pensat per automatitzar i millorar la gestió de les competicions que organitza la Federació Catalana de Rem (FCR).

El sistema, per una banda, permetrà als espectadors gaudir en temps real del desenvolupament de la competició. Això vol dir que el públic en tot moment podrà saber quines regates[1] estan a punt de començar, quines estan en curs i quines finalitzades. Òbviament també disposaran d'informació sobre els participants de cadascuna d'aquestes regates. Val a dir també que aquesta informació serà presentada al públic en un format molt intuïtiu que qualsevol persona, tingui coneixements de rem o no, podrà entendre.

Per altra banda, Iris també disposa d'una aplicació web on es poden consultar l'històric de totes les competicions realitzades fins al moment. Aquesta web és d'ús públic i qualsevol hi pot accedir.

Finalment, i degut a requeriments de la FCR, la base de dades d'Iris està implementada de tal manera que pugui ser modificada tant a través de crides a un WebService com directament per personal de la federació. I directament vol dir sense haver de fer consultes SQL, ja que aquesta base de dades es pot visualitzar i gestionar com un full de càlcul que, de fet, és del que es tracta.

1.2.- Motivació

Fins al moment la FCR gestiona les regates de la següent manera:

1.- Hi ha un àrbitre a la sortida (que l'anomenarem AS) i un a l'arribada (AA) amb un *walkie talkie* cadascun, a més AA té un crono.

2.- Quan s'inicia la regata AS diu quelcom pel *walkie talkie* i AA inicia el crono, òbviament amb un cert retard.

3.- Cada cop que arriba una embarcació d'un participant, AA llegeix el temps d'arribada i el guarda.

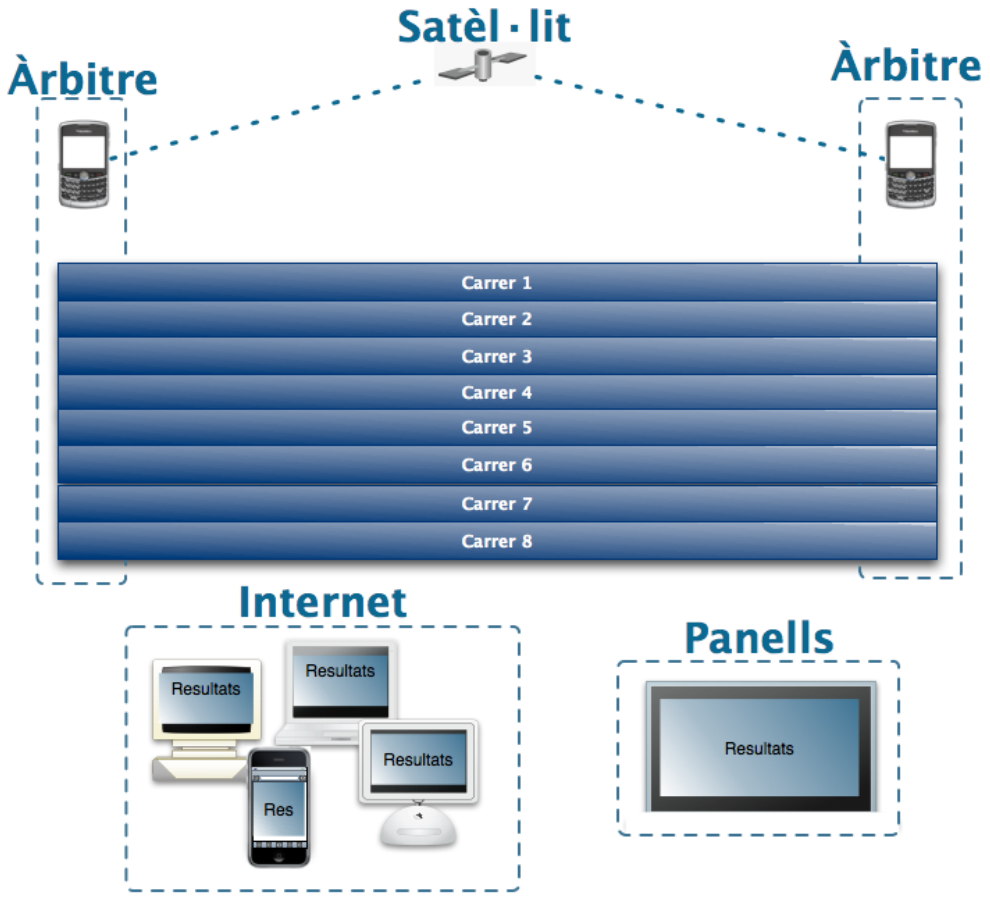
4.- Un cop finalitzada la regata AA porta els temps a la persona encarregada d'emmagatzemar-los a un full de càlcul el qual, al mateix temps, es mostra per una pantalla.

Com es pot observar, aquesta manera de procedir pot provocar (i de fet provoca) pèrdues d'informació i de precisió. A més a més no dóna informació als espectadors en temps real, sinó que fins una bona estona després d'haver acabat la regata, els espectadors no saben els resultats.

La motivació d'Iris és doncs evitar, en la mesura del possible, aquesta pèrdua d'informació i de precisió automatitzant aquest escenari descrit anteriorment. A més es vol oferir als espectadors suport visual i en temps real de l'estat de la competició.

1.3.- Descripció del projecte

Ja s'ha fet una explicació a grans trets de què es vol aconseguir amb aquest projecte. En aquest apartat entrarem una mica més en matèria i realitzarem una explicació més detallada de què es vol aconseguir.



Imatge 1.1: Esquema representatiu del sistema

La figura 1.1 mostra una representació del que vol arribar a ser el sistema. Es poden identificar clarament les diverses parts en les que es divideix Iris.

Per una part es volen introduir PDAs[2] com a substitut del cronòmetre convencional que s'utilitza actualment a la FCR. Aquestes PDAs estaran comunicades en tot moment amb satèl·lits GPS[3], la qual cosa els permetrà poder realitzar cronometratges sobre un temps unívoc per a tothom, el temps GPS. A més a més, es comunicaran directament amb un servidor que s'encarregarà d'emmagatzemar els resultats obtinguts als fulls de càlcul esmentats anteriorment, que fan la funció de base de dades.

A part d'això tenim la pantalla informativa de l'estat de la competició. Aquesta pantalla mostra les regates que començaran en un breu període de temps, les regates ja finalitzades amb les dades de cada participant (posició final, tripulants, temps,...) i per últim les regates que hi ha en curs. Aquesta pantalla s'anirà comunicant periòdicament amb el servidor per descarregar-se i mostrar les dades actuals de la competició.

Cal dir també que en qualsevol moment es poden modificar les dades del full de càlcul manualment i aquestes es veuran reflectides al servidor en un breu període de temps. Això és necessari per a la FCR per diversos aspectes que veurem més endavant.

Finalment tenim la pàgina web que mostra l'històric de competicions. Aquest servei consta, per una part, d'un contingut històric (que podríem anomenar estàtic) de totes les competicions acabades. Però això no és tot, i és que suposant que ens trobem en un dia de competició, aquesta es va mostrant en temps quasi real a la web. En altres paraules, qualsevol persona podrà consultar l'estat de la competició des de casa sense haver de ser present al lloc on s'estan realitzant les regates. Aquesta web també s'actualitzarà automàticament a partir de les dades que es van recollint al servidor.

1.4.- Objectius

L'objectiu principal que intenta assolir el projecte Iris és:

Millorar i automatitzar la gestió de les competicions de la Federació Catalana de Rem.

Tot i això, aquest objectiu es pot desglossar en una sèrie de subojectius que s'esmenten a continuació:

1.- Introduir les PDAs amb un software a mida com a eina per al cronometratge de les regates reemplaçant els simples cronòmetres utilitzats fins al moment.

2.- Aprofitar la potència de les PDAs per evitar el màxim possible els errors i/o desviacions en les dades que s'obtenen dels cronometratges.

3.- Donar un sistema que permeti la modificació de les dades tant per part de WebServices[4] cridats per les PDAs com directament per un usuari.

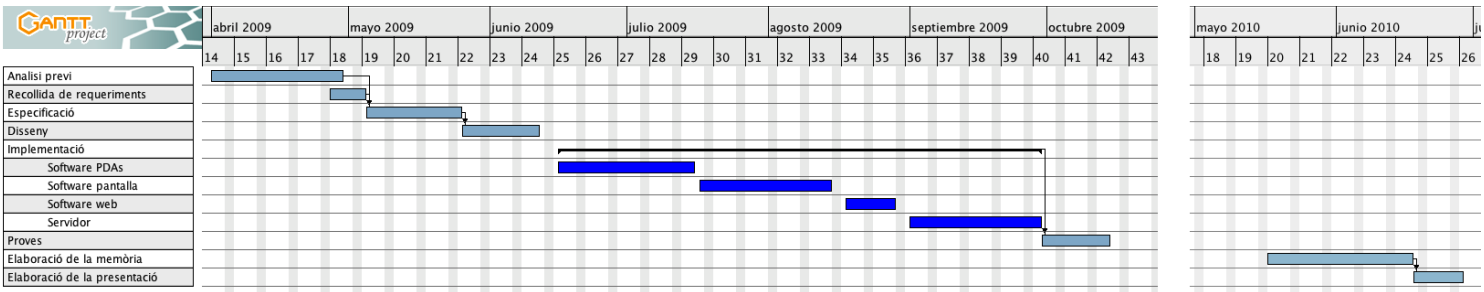
4.- Automatitzar la recollida i posterior gestió de les dades (temps de sortida, temps d'arribada, crono, posicions finals,...) dels diferents participants de cadascuna de les regates.

5.- Donar suport visual als espectadors de les competicions mitjançant una pantalla que, en temps real, mostrarà l'evolució de la competició (regates pendents, en curs i finalitzades).

6.- Donar suport web per poder visualitzar, per una banda, l'històric de competicions que s'han realitzat fins al moment i, per altra banda, l'estat de la competició actual si n'existeix alguna.

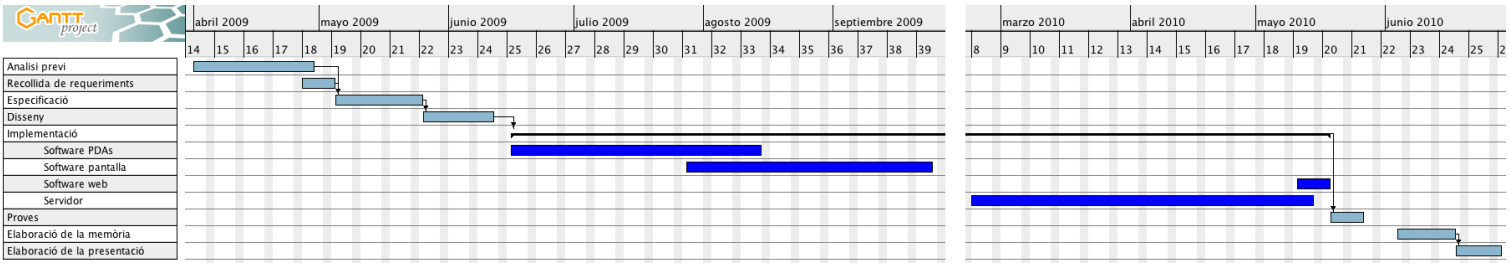
1.5.- Planificació

La Imatge 1.1 mostra la planificació que es va realitzar abans de començar el projecte. Com es pot observar, es va pensar que la part que ocuparia més temps seria la d'implementació. Això es deu a què qui havia d'implementar el projecte havia de documentar-se sobre diverses tecnologies. Ja que, com es veurà, cadascuna de les parts a implementar utilitza una tecnologia diferent.



Imatge 1.2: Diagrama de Gantt planificació

Finalment, i com era d'esperar, les dates no van anar tal com s'esperava. Tot i que l'inici del projecte sí que es va desenvolupar amb total normalitat, no va passar el mateix amb la part crítica, la implementació. Com es pot observar a la figura 1.2 va augmentar el temps d'implementació del software de les PDAs i del servidor, així com de la pantalla on es van tenir certs problemes que s'explicaran en el seu moment. Contràriament el temps d'implementació de l'aplicació web va disminuir mínimament.



Imatge 1.3: Diagrama de Gantt real

A més a més, podem veure que el projecte va estar aturat des de mitjans de setembre del 2009 fins a finals de febrer de 2010. Això és degut a que, a causa de prioritats de l'empresa, qui implementava el projecte es va haver de dedicar a un projecte diferent.

1.6.- Anàlisi econòmica

A continuació es farà una valoració econòmica aproximada del projecte. Aquesta valoració es dividirà en 3 parts: per una banda analitzarem les llicències de software, eines de programació i quotes de servidors. Posteriorment valorarem el hardware necessari i finalment la mà d'obra utilitzada.

Producte	***Preu***
OpenOffice	0,00 €
Eclipse	0,00 €
GanttProject	0,00 €
Gimp	0,00 €
OmniGraffle	0,00 €
Java	0,00 €
Java FX	0,00 €
C# i .Net	0,00 €
MySQL	0,00 €
Windows mobile	100,00 €
Google docs (Spreadsheets)	0,00 €
Cost Total	100,00 €

Taula 1.1: Cost llicències i eines de programació

Com es pot observar a la taula 1.1 tot el software que s'ha utilitzat pertany a programari lliure, per tant no aporta costos extra al projecte.

Producte	*Quantitat*	*Preu/unitat*
PDA	2	600,00 €
Pantalla HD	1	800,00 €
Pc per alimentar pantalla	1	500,00 €
GPS de Pc de la pantalla	1	50,00 €
Caixa protectora Pc i pantalla	2	200,00 €
Quota servidor allotjament Iris	1	20,00€/mes
Internet mòbil 3G	1	30,00 €
Interfície gràfica pàgina web	1	200,00 €
Interfície gràfica pantalla	1	150,00 €
Cost Total (1 any de lloguer de servidor)		3.570,00 €

Taula 1.2: Cost hardware

Com mostra la taula 1.2 Iris té una part més o menys important de hardware que farà augmentar el preu final del producte.

Etapes del projecte	*Perfil*	*Hores*	*Preu/hora*	*Cost*
Anàlisi prèvia	Cap projecte	168	50	8.400,00 €
Recollida de requeriments	Cap projecte	24	50	1.200,00 €
Especificació	Analista	120	40	4.800,00 €
Disseny	Analista	104	40	4.160,00 €
Implementació	Programador	912	24	21.888,00 €
Proves	Programador	40	40	1.600,00 €
Documentació	Analista	10	40	400,00 €
Total		**1378**		**42.448,00 €**

Taula 1.3: Mà d'obra

Tot i el que mostra la taula 1.3, gran part d'aquest projecte ha estat portat a terme com a conveni de cooperació educativa, és a dir, els salaris que mostra la taula en cap cas han estat els reals. L'única finalitat d'aquesta taula era mostrar el preu real aproximat d'un projecte d'aquesta envergadura.

1.7.- Organització de la memòria

Aquesta memòria està organitzada de la següent manera:

L'apartat en el qual ens trobem intenta mostrar una visió general de l'abast d'Iris. Això és definir què és el que engloba exactament aquest projecte. Tot i això ho fa amb una visió general sense entrar en detalls de com es realitzarà el projecte. A més a més aquest apartat també mostra la planificació seguida per a la realització de les tasques. I finalment mostra el cost econòmic aproximat que costaria.

El següent apartat mostrarà quin és el punt de partida del projecte, és a dir, quina era la manera de procedir per part de la FCR pel que fa al cronometratge de les regates. A més es realitzarà una anàlisi de quines són les mancances que té el sistema actual de cronometratge de regates utilitzat per la FCR. Finalment s'explicarà un sistema d'exemple que s'utilitza al cronometratge de regates als Jocs Olímpics.

Al tercer apartat es mostra la recollida de requeriments que es va fer. Aquests requeriments es van decidir després de tenir diverses reunions amb els responsables de la FCR i escoltar quines eren les seves necessitats i prioritats en quant a l'aplicació que se'ls havia de proporcionar.

Un cop feta la recollida de requeriments és moment de formalitzar totes aquestes necessitats en una especificació formal. En aquesta especificació es mostraran tots els casos d'ús que tenen a veure amb les diverses parts de l'aplicació.

Com és d'esperar, ara cal fer el pas de què es vol fer, que és el que s'ha formalitzat a l'apartat anterior, a com s'ha de fer. Per tant en aquest punt es mostrarà com ha estat dissenyat Iris. Aquí, a més, es donarà una visió gràfica de l'arquitectura del sistema perquè el lector es familiaritzi amb la forma que tindrà l'aplicació. A més a més s'explicaran les diferents alternatives que s'han tingut en compte a l'hora de dissenyar el producte.

Posteriorment s'explicaran les tecnologies utilitzades a l'hora d'implementar el sistema. A més, s'explicaran certs canvis que s'han realitzat al moment d'implementar ja que no s'havien previst al dissenyar l'aplicació, però són clarament necessaris per al correcte funcionament d'aquesta.

Un cop acabada la implementació és moment d'integrar les diverses parts en les quals està dividit Iris i realitzar proves exhaustives per corroborar el seu correcte funcionament.

Finalment es mostraran els productes obtinguts. En aquest apartat es mostren captures de pantalla acompanyades d'explicacions de cadascun dels softwares desenvolupats.

2.- ANÀLISI PREVIA

2.1.- Punt de partida

Al punt 1.2 s'explica la metodologia que segueix actualment la FCR per al cronometratge de les regates. Com es pot comprovar aquest sistema és bastant precari.

Després d'assistir a algunes d'aquestes competicions i realitzar-ne una anàlisi, s'ha arribat a la conclusió que les mancances que té el sistema actual són les següents (recordem que AS=Àrbitre sortida i AA=Àrbitre arribada):

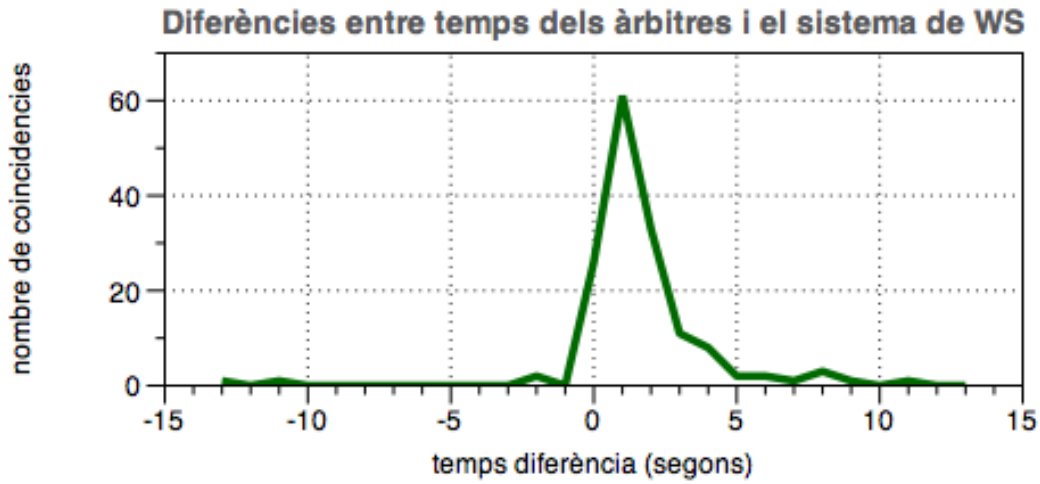
1.- Els temps que va enregistrant AA poden estar esbiaixats ja que, en principi, aquest no té contacte visual amb les embarcacions al moment de la sortida. A més fins que AS no li dona l'avís i AA posa el cronòmetre en funcionament poden passar uns quants segons.

2.- Els espectadors de la regata no tenen informació d'una determinada regata fins passat un temps més o menys llarg del final de la regata. En cap cas tenen informació de la regata mentre aquesta està en curs.

3.- La informació que es proporciona al públic, a més de ser tardana, és poc clara, ja que l'únic que poden observar aquests espectadors és un full de càlcul on un empleat va introduint manualment els temps que realitzen els participants.

4.- Qualsevol persona que no es trobi al lloc on s'està realitzant la regata, no té manera de saber com s'està desenvolupant aquesta.

5.- Actualment no hi ha manera que qualsevol pugui consultar l'històric de competicions a través del seu propi ordinador.



Imatge 2.1: Diferències entre temps dels àrbitres i el sistema de WS

El que mostra la imatge 2.1 fa referència al punt 1 explicat anteriorment. És la distribució normal del nombre de coincidències envers al temps. En altres paraules, al temps 0 es mostren el nombre de vegades que ha coincidit el temps dels àrbitres amb el del nostre sistema. Al temps 1 es mostren el nombre de vegades que els àrbitres s'han retardat 1 segon envers el nostre i així successivament.

Si el sistema actual fos encertat el gràfic hauria d'estar centrat a 0, però en canvi està centrat a 1. Això corrobora el que es deia al punt 1, que degut a retards en el sistema de cronometratges actual, els temps obtinguts no són del tot correctes.

2.2.- Estudi del sistema utilitzat als Jocs Olímpics

Swiss Timing és el nom d'una empresa que es dedica al cronometratge i suport a la visualització de diferents esports, entre els quals es troba el rem. De fet, Swiss Timing és l'empresa que s'encarrega de les proves de rem que es realitzen durant els Jocs Olímpics. Per aquest motiu es va creure convenient realitzar un estudi de la seva manera de procedir. D'aquesta manera es disposaria d'un referent a l'hora de dissenyar Iris.

Cal remarcar però, dos aspectes importants pel que fa a aquesta empresa. Per una banda, Swiss Timing ofereix un servei que és contractat pels organitzadors dels Jocs Olímpics, mentre que Iris és un producte que es desenvolupa per a la FCR. El fet de ser un servei implica que, a part del hardware necessari, Swiss Timing també proporciona els recursos humans necessaris, mentre que la FCR haurà de disposar del seu propi personal.

Per altra banda, un altre fet remarcable és el pressupost del qual disposa aquesta empresa, que és immensament superior al pressupost que té la FCR. Al mateix temps, els resultats obtinguts a les regates dels Jocs Olímpics han de tenir una fiabilitat molt més elevada que a nivell de Catalunya. Això no vol dir, en cap cas, que els resultats que s'obtenen a la FCR no siguin correctes.

En resum, Iris no té la finalitat de plagiar la manera de funcionar de Swiss Timing, però es va creure convenient realitzar una anàlisi per extreure'n algunes idees.

El que fa aquesta empresa és el següent:

1.- Primerament s'instal·la un emissor GPS i un acceleròmetre a cadascuna de les embarcacions.

2.- La pistola que dispara el tret de sortida, està connectada al cronòmetre que enregistrarà els temps dels participants. És a dir, quan es realitza la sortida, es posa el cronometre en funcionament automàticament.

3.- Cada 500 metres es realitza un *checkpoint*[5] on hi ha una persona per cada embarcació que participa a la regata. Cadascuna d'aquestes persones té assignada una embarcació i, quan aquesta passa pel punt de control realitzen un *split*[6]. D'aquesta manera es poden saber les diferències de temps entre les embarcacions.

4.- L'emissor GPS que porta cada embarcació és utilitzat (amb un cert retard) per saber en quin punt es troba cada embarcació en un moment donat. D'aquesta manera es pot crear un esquema de com s'està desenvolupant la regata.

5.- A l'arribada es realitza el mateix procediment que al punt 3.

6.- Tant al punt 3 com al 5, el sistema emmagatzema el temps de cada participant i el mostra al públic en ordre d'arribada.

7.- L'acceleròmetre que s'esmenta a l'apartat 1 s'utilitza per a extreure estadístiques sobre l'acceleració, velocitat i balanceig de les embarcacions.

3.- RECOLLIDA DE REQUERIMENTS

A continuació es mostren el seguit de requeriments que mostra Iris, és a dir, quines característiques ha de tenir el producte a desenvolupar. Cadascun dels subapartats mostrarà les característiques que ha de tenir una part del sistema.

Cal dir que la majoria de requeriments han estat establerts per part de la FCR. Tot i això, la nostra empresa WorldSensing, i en particular Jordi Llosa (que ha estat involucrat en la federació diversos anys) ha fet valer la seva experiència personal per tal d'aportar el seu gra de sorra i intentar aconseguir un producte millor.

3.1.- Representació de la informació

3.1.1.- Com es representa la informació

Una de les exigències que plantejava la FCR era que la representació de la informació havia de ser fàcilment modificable per part d'un usuari. Això és així per diversos motius, encara que per poder entendre'ls és necessari explicar primer com es genera i representa actualment la informació de les competicions:

La federació disposa d'un empleat encarregat de crear les regates pertanyents a una competició. La seva funció és crear un full de càlcul on insereix manualment totes les regates. Com es pot observar a la imatge 3.1, cada regata consta d'unes dades genèriques pròpies com són el nom, la modalitat i l'hora programada d'inici. A més també conté la llista de participants d'aquesta amb les dades de cadascun com són el carrer, el club, els remers i el temps. Aquest últim camp es deixa en blanc ja que serà omplert una vegada realitzada la regata.

	A	B	C	D	E	F	G	H	I	J
1	Mànega 1		2xAM 500M	H. 11:00			Mànega 6		1xIM 2º 1000M	H. 11:25
2	Carrer	Club	Remers	Temps			Carrer	Club	Remers	Temps
3	1	CNA	Toni Bo - Alex Pagà				1	CNB 5	GABRIEL LUCAS	
4	2	RCNT 1	Marc Ribot - Alvaro Romero				2	CNA 1	Alex Llor	
5	3	CNS	Gabriel Sánchez- Joel Valor				3	CRT 1	JORDI ANDREU	
6	4	CNB 1	Jesus Corominas-Ahron Inocent				4	RCMB 1	Enric Florit	
7	5	CCM 1	RAMÓN BEÁ- OSCAR HERNÁNDEZ				5	CNA 5	Ximo Monllau	
8	6	CCM 2	CHRISTIAN MORALES-FRAN MONTULL				6	CRT 4	ROGER FOLQUE	
9										
10	Mànega 2		2xAM 500M	H. 11:05			Mànega 7		1xIM 2º 1000M	H. 11:30
11	Carrer	Club	Remers	Temps			Carrer	Club	Remers	Temps
12	1	CRT 1	ALEX SORRIBES, GERARD GARCIA				1	CRT 2	JAUME FABREGAT	
13	2	CNB 2	Eduard Gelí- Oriol Masó				2	CNB 3	JOSEP MASÓ	
14	3	RCMB	Gerard Andreu - Martí Coy				3	CCM	JACOBO NICOLAU	
15	4	CCM 3	MIGUEL RUBIO-MARC ROCA				4	CNA 4	Aniol Vidal	
16	5	RCNT 2	Aleix Voltes - David Lorente				5	CNB 1	DAVID PLANES	
17	6	CRT 2	ISMAEL FORES, ENRIC				6	CRT 6	JAVI SANTIRSO	

Imatge 3.1: Representació actual de la informació d'una competició

Una vegada entès com es representen les competicions, passem a explicar els motius d'aquest requeriment per part de la federació.

El principal motiu és que sovint es produeixen situacions que no s'havien previst en anterioritat com poden ser, per exemple, l'absència d'algun dels participants d'una regata, o que algun dels participants no acabi la regata, o fins i tot que a última hora s'afegeixi un nou participant que no estava a la llista inicial. Òbviament, per tal de no endarrerir en excés l'inici de la regata, això ha de ser fàcil i ràpid de plasmar al lloc on es representa la informació.

L'altre motiu és que l'empleat encarregat de generar les competicions ha de continuar realitzant la seva feina de la manera més semblant possible a l'actual per evitar haver de formar-lo en altres metodologies.

3.1.2.- Detecció d'errors a la representació de les dades

Com es pot veure, aquest sistema d'emmagatzemar la informació dóna rapidesa per a fer modificacions, però per contra té grans inconvenients. Per tant, també serà un requeriment trobar una manera de representar la informació que eviti els inconvenients explicats a continuació.

Primerament és molt fàcil, sobretot en competicions de gran volum de regates, cometre errors com podrien ser: deixar una competició a mitges, repetir noms de competicions, posar noms de participants o clubs erronis, etc.

És per això que es vol buscar un sistema que ajudi l'usuari a detectar aquests errors a priori i poder-los solucionar. Aquest sistema vindria a ser un *parser*[7] de les dades. És a dir, una aplicació que aniria recorrent totes les regates d'una competició i comprovaria que tots els camps fossin correctes.

3.1.3.- Interacció amb les dades

Ja s'ha vist a la figura 3.1 que les dades es representen en un simple full de càlcul. Aquest fet fa que sigui impossible interactuar amb les dades a través de sistemes que no es trobin a la mateixa màquina en la qual estan les dades (sistemes distribuïts).

Aquest tipus de representació provoca un gran desavantatge pel que fa a la manera com es té previst muntar el sistema ja que, com s'ha dit anteriorment, es vol automatitzar la gestió de les competicions. I aquesta automatització també implicarà interconnexions entre diversos sistemes que es troben en diferents llocs físics.

3.1.4.- Històric de competicions

Un altre requeriment que plantejava la FCR era generar un històric de competicions que puguin ser consultades per qualsevol persona independentment d'on es trobi.

A més a més disposant d'aquesta informació acumulada es poden treure una gran varietat d'estadístiques que també es poden presentar als usuaris. Aquestes estadístiques es podrien classificar segons siguin:

Estadístiques a nivell de remer o de clubs:

1.- Nombre de competicions en les que ha participat.

2.- Número de victòries aconseguides.

3.- Millors temps realitzats.

4.- Número de podis aconseguits.

...

Estadístiques a nivell de federació:

1.- Clubs o remers amb més participacions.

2.- Clubs o remers amb més podis.

3.- Millors temps realitzats per modalitat.

4.- Nombre de competicions realitzades.

...

3.1.5.- Opcions disponibles

Una vegada estudiats els diferents requeriments que ha de tenir el sistema en quant a la representació de la informació, és moment de veure les diferents opcions que existeixen per guardar i gestionar aquesta informació.

Opció	Pros	Contres
Fulls de càlcul	+ Funcionament senzill i exactament igual a l'actual. + Rapidesa a l'hora d'inserir informació que no havia estat prevista anteriorment. + Possibilitat de ser atacat per un software de detecció d'errors.	- Facilitat d'obtenir dades corruptes. - Impossible interactuar amb les dades de manera remota. - Dificultat per gestionar l'històric de competicions. - La informació està emmagatzemada de manera local.
Base de dades convencional	+ Possibilitat d'interactuar amb les dades de manera remota. + Impossible d'inserir dades corruptes si s'ha fet un bon disseny de la base de dades. + Facilitat a l'hora de guardar un històric de competicions i consultar-les quan sigui necessari.	- Necessitat d'una aplicació per a gestionar-la. - Usuari poc familiaritzat amb bases de dades.
Spreadsheets[8]	+ Funcionament molt semblant a l'actual. + Rapidesa per inserir informació que no havia estat prevista. + La FCR no s'ha de preocupar de guardar la informació, aquesta estarà a GoogleDocs[9]. + Existència d'una API[10] de Google que facilita l'extracció de la informació. + Facilitat de guardar l'històric de competicions.	- Temps d'execució elevant en competicions de gran volum de dades. - Temps d'execució elevat a l'hora d'aconseguir l'històric de competicions.

Una vegada analitzades totes les possibilitats s'ha decidit utilitzar els Spreadsheets de Google com a sistema per emmagatzemar la informació.

Els motius d'aquesta elecció són tots els que es mostren a la columna de pros. Tot i això, el motiu que més ha fet decantar la balança cap aquesta opció ha estat l'existència de l'API de Google, que ens proporciona serveis per interactuar amb l'Spreadsheet de manera simple.

No obstant, s'ha de tenir present que aquesta API algunes vegades té un temps de resposta bastant elevat, i això és perjudicial per a la nostra aplicació. Per tant en cas que quan s'implementi l'aplicació es doni aquesta situació, s'haurà de buscar alguna solució alternativa.

3.2.- Presentació de la informació al públic

La FCR vol proporcionar informació al públic de les competicions que han realitzat en el passat i que s'estan realitzant en un moment donat. Òbviament no serveix el mateix sistema per a mostrar les competicions actuals (sovint hi ha canvis, hi ha regates pendents, en curs i acabades, el públic es troba concentrat al lloc on s'esta realitzant la competició) que per a representar les passades (no hi ha un lloc físic on es trobi tothom que vulgui consultar les competicions, les regates estan totes finalitzades, no hi ha canvis en les regates, etc).

3.2.1.- Presentació de les competicions acabades

Pel que fa a les competicions acabades, la FCR requereix un sistema amb el qual les persones relacionades amb la pròpia federació i, per què no, qualsevol altra persona interessada en el tema, puguin consultar aquesta informació.

El que més es valora en aquesta part és que no s'hagi d'anar a buscar aquestes dades a cap lloc físic, sinó que es puguin consultar des de qualsevol lloc. En altres paraules, s'està demanant una pàgina web. A més, la FCR valora positivament que es presentin les dades de manera vistosa i entenedora per tothom, ja que així poden captar l'atenció de gent que, a priori, no està interessada en aquest esport.

Després de discutir-ho amb la federació i per problemes temporals i econòmics s'ha decidit, de moment, que la web només mostrarà les dades de les competicions. És a dir, la funcionalitat de filtrar la informació per participant o club així com de generar estadístiques queda pendent per a una segona versió del producte.

3.2.2.- Presentació de les competicions en curs

Les competicions en curs es volen mostrar al públic assistent per mitjà d'una pantalla on tant aquest públic com els propis participants puguin anar veient com s'està desenvolupant la competició.

Aquesta informació, a diferència de com s'està fent actualment, s'ha de mostrar en temps real i d'una manera vistosa i clara. Encara més que la web que s'esmentava a l'apartat anterior ja que ha de donar una sensació de dinamisme.

Quan una competició s'està realitzant, les regates van canviant d'estat. Primerament estan pendents, una vegada iniciades passen a estar en curs i s'haurà de visualitzar com va corrent el temps (per a donar aquesta sensació de 'moviment'). Finalment passaran a l'estat d'acabades on es mostraran les classificacions dels diversos participants amb els respectius temps finals.

Finalment també s'ha cregut convenient que les competicions en curs es puguin mostrar a la web de competicions acabades. D'aquesta manera algú que no estigui físicament al lloc on s'està realitzant la competició, podrà anar seguint també el desenvolupament d'aquesta, encara que sigui amb el format de les competicions finalitzades.

3.3.- Cronometratge de les regates

En referència a la manera de cronometrar les regates, s'ha decidit suplantar la manera actual de procedir per la seva precarietat i baixa fiabilitat.

Com s'ha comentat ja diverses vegades durant aquesta memòria, es vol automatitzar el sistema, per tant és necessari introduir un hardware bastant més potent que els simples cronòmetres utilitzats actualment.

Aquest hardware haurà de ser capaç de comunicar-se amb les dades de la competició per a enviar i rebre informació. A més, donat que als llocs on s'utilitzarà no es disposarà segurament de cobertura Wifi, serà necessari que disposi de connexió GPRS[11] per poder comunicar-se amb les dades. I òbviament haurà de ser un hardware on es puguin instal·lar i executar aplicacions a mida.

Després de realitzar un estudi de quins dispositius serien més útils per a dur a terme aquesta tasca i amb aquests requisits, s'ha arribat a la conclusió que la millor elecció és utilitzar PDAs. Ja que a part de complir amb tots els requisits esmentats, són fàcilment portables i es poden desenvolupar aplicacions amb les quals l'usuari pot interactuar a través de simples clics a la pantalla.

3.4.- Requeriments genèrics

En aquest apartat s'expliquen aquells requeriments que, tot i que no han estat especificats directament per la FCR, sí que s'han considerat importants pel que fa a WorldSensing i a l'obtenció d'un producte ben acabat.

Primerament, pensant en promocionar Iris no tan sols a altres federacions de rem, sinó també a altres esports que tinguin característiques semblants a les esmentades anteriorment, l'empresa WorldSensing vol desenvolupar el producte de la manera més genèrica possible. Amb això es vol aconseguir poder implantar aquest producte a altres àmbits realitzant el mínim de canvis possibles al programari.

En segon lloc, s'ha pensat en la fiabilitat i la robustesa de l'aplicació. La major part de software i hardware informàtic (per no dir tot) no és perfecte al 100%, això vol dir que, en determinades situacions i per diverses circumstancies, poden no obtenir-se els resultats esperats. No obstant, ens trobem en un escenari, sobretot pel que fa a la part de les PDAs, que no hauria de fallar en cap moment ja que ralentitzaria el desenvolupament de la competició i podria donar una mala imatge del producte. Per això s'intentarà obtenir una aplicació que doni resposta a qualsevol error que es pugui donar. Algun exemple podrien ser: la pèrdua de cobertura GPRS per part de les PDAs, que la pantalla en un moment donat no pugui connectar al servidor, etc. Desafortunadament però, sempre existeixen situacions imprevisibles, tot i això s'intentarà reduir-les al màxim.

4.- ESPECIFICACIÓ

4.1.- Model de casos d'ús

Un cas d'ús és un document que descriu una seqüència d'esdeveniments que realitza un actor que fa us del sistema per a alguna finalitat.

Per la seva banda, un actor és una entitat externa al sistema que participa en el desenvolupament del cas d'ús. Un actor pot ser una persona, conjunt de persones un sistema hardware/software,...

Una vegada extrets els requeriments del sistema i definida l'arquitectura que ha de tenir, el que cal fer ara és definir els casos d'ús que es deriven dels requeriments anteriors. Els agruparem segons la part del sistema on s'utilitzaran.

4.1.1.- Casos d'ús aplicació PDA

4.1.1.1.- Seleccionar competició

Diagrama de casos d'ús

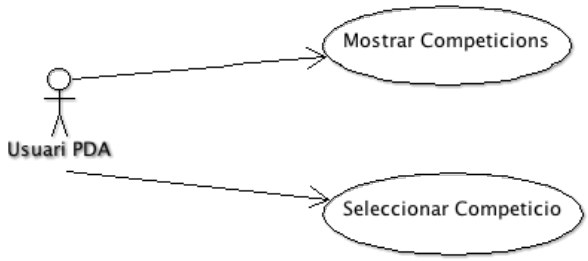


Diagrama 4.1: Diagrama de casos d'ús Seleccionar competició

Especificació dels casos d'ús

CAS D'ÚS	**Mostrar Competicions**
ACTORS	Usuari PDA
PROPÒSIT	Llistar les competicions existents al sistema
RESUM	L'actor accedeix a l'aplicació i automàticament aquesta li retorna el llistat de competicions

Taula 4.1: Cas d'ús Mostrar Competicions

Usuari	***Sistema***
1.- L'usuari accedeix a l'aplicació	
	2.- El sistema envia una petició al servidor per aconseguir totes les competicions i en mostra un llistat

Taula 4.2: Curs típic d'esdeveniments Mostrar Competicions

CAS D'ÚS	**Seleccionar Competició**
ACTORS	Usuari PDA
PROPÒSIT	Seleccionar una de les competicions que ha llistat el sistema anteriorment
RESUM	L'usuari selecciona una de les competicions i el sistema la carrega

Taula 4.3: Cas d'ús Seleccionar Competició

Usuari	***Sistema***
1.- L'usuari selecciona una de les competicions disponibles	
	2.- El sistema carrega aquesta competició i mostra la vista de selecció de rol

Taula 4.4: Curs típic d'esdeveniments Seleccionar Competició

4.1.1.2.- Seleccionar rol àrbitre

Diagrama de casos d'ús

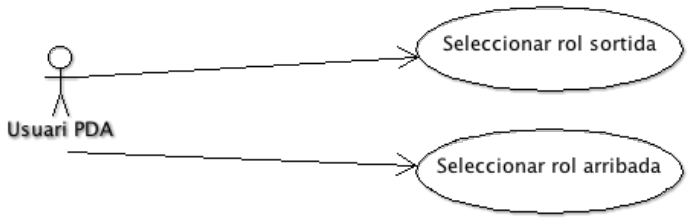


Diagrama 4.2: Diagrama de casos d'ús Selecció rol àrbitre

Especificació dels casos d'ús

CAS D'ÚS	**Seleccionar rol sortida**
ACTORS	Usuari PDA
PROPÒSIT	Preparar l'aplicació per a ser àrbitre de sortida
RESUM	L'actor selecciona ser àrbitre de sortida i el sistema carrega la vista d'àrbitre de sortida

Taula 4.5: Cas d'ús Seleccionar rol sortida

Usuari	***Sistema***
1.- L'usuari selecciona ser àrbitre de sortida	
	2.- El sistema mostra la vista corresponent a l'arbitratge de sortida

Taula 4.6: Curs típic d'esdeveniments Seleccionar rol sortida

CAS D'ÚS	**Seleccionar rol arribada**
ACTORS	Usuari PDA
PROPÒSIT	Preparar l'aplicació per a ser àrbitre d'arribada
RESUM	L'actor selecciona ser àrbitre d'arribada i el sistema carrega la vista d'àrbitre d'arribada

Taula 4.7: Cas d'ús Seleccionar rol arribada

Usuari	*Sistema*
1.- L'usuari selecciona ser àrbitre d'arribada	
	2.- El sistema mostra la vista corresponent a l'arbitratge d'arribada

Taula 4.8: Curs típic d'esdeveniments Seleccionar rol arribada

4.1.1.3.- Seleccionar regata

Diagrama de casos d'ús

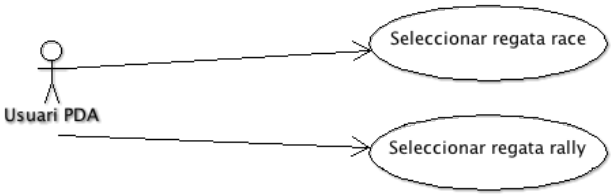


Diagrama 4.3: Diagrama de casos d'ús Seleccionar regata

Especificació dels casos d'ús

CAS D'ÚS	**Seleccionar regata race**
ACTORS	Usuari PDA
PROPÒSIT	Mostrar la vista de regata tipus *race* a la PDA
RESUM	L'actor selecciona una regata tipus *race* i el sistema carrega la vista corresponent de la regata.

Taula 4.9: Cas d'ús Seleccionar regata race

Usuari	*Sistema*
1.- L'usuari selecciona una de les regates tipus *race* del conjunt de regates	
	2.- El sistema mostra la vista tipus *race* (de sortida o d'arribada, depenent del rol seleccionat anteriorment) d'aquesta regata.

Taula 4.10: Curs típic d'esdeveniments Seleccionar regata race

CAS D'ÚS	**Seleccionar regata rally**
ACTORS	Usuari PDA
PROPÒSIT	Mostrar la vista de regata tipus *rally* a la PDA
RESUM	L'actor selecciona una regata tipus *rally* i el sistema carrega la vista corresponent de la regata.

Taula 4.11: Cas d'ús Seleccionar regata rally

Usuari	***Sistema***
1.- L'usuari selecciona una de les regates tipus *rally* del conjunt de regates	
	2.- El sistema mostra la vista tipus *rally* (de sortida o d'arribada, depenent del rol seleccionat anteriorment) d'aquesta regata.

Taula 4.12: Curs típic d'esdeveniments Seleccionar regata rally

4.1.1.4.- Insertar/Anul·lar SplitTime

Diagrama de casos d'ús

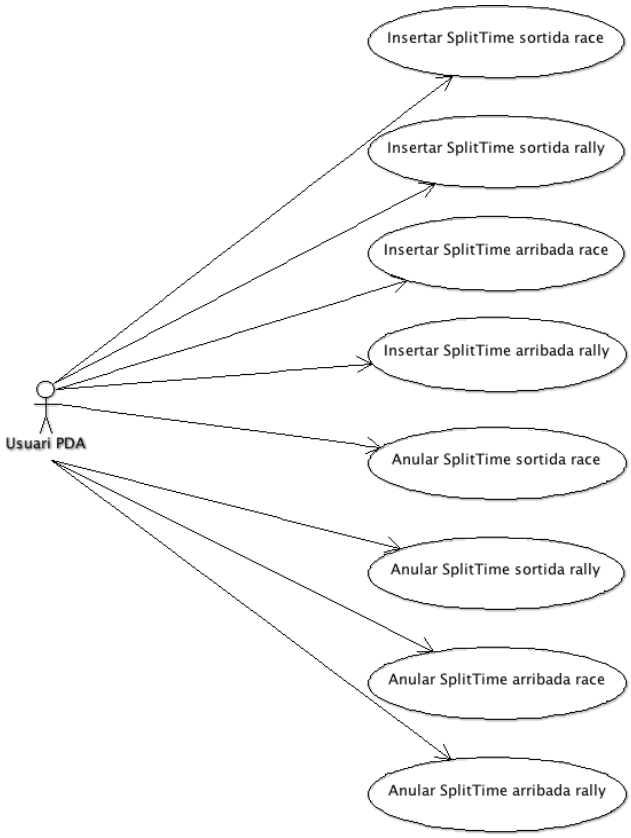


Diagrama 4.4: Diagrama de casos d'ús Insertar/Anul·lar SplitTime

Especificació dels casos d'ús

CAS D'ÚS	**Insertar SplitTime sortida race**
ACTORS	Usuari PDA
PROPÒSIT	Iniciar una regata *race*
RESUM	L'usuari inicia una regata tipus *race*

Taula 4.13: Cas d'ús Insertar SplitTime sortida race

Usuari	*Sistema*
1.- L'usuari prem el botó iniciar regata de la vista de sortida d'una regata *race*	
	2.- El sistema captura l'hora GPS i envia la petició d'inici de regata
3.- L'aplicació mostra la regata iniciada amb el cronòmetre corresponent	

Taula 4.14: Curs típic d'esdeveniments Insertar SplitTime sortida race

CAS D'ÚS	**Insertar SplitTime sortida rally**
ACTORS	Usuari PDA
PROPÒSIT	Iniciar una regata *rally*
RESUM	L'usuari inicia una regata tipus *rally*

Taula 4.15: Cas d'ús Insertar SplitTime sortida rally

Usuari	*Sistema*
1.- L'usuari selecciona un dels participants de la llista i prem el botó iniciar regata.	
	2.- El sistema captura l'hora GPS i envia la petició d'inici de regata juntament amb l'id del participant afectat.
3.- L'aplicació mostra la llista de participants indicant quins han iniciat la regata i quins encara no.	

Taula 4.16: Curs típic d'esdeveniments Insertar SplitTime sortida rally

CAS D'ÚS	**Insertar SplitTime arribada race**
ACTORS	Usuari PDA
PROPÒSIT	Fer una arribada d'un participant d'una regata *race*
RESUM	L'usuari enregistra al sistema una arribada d'un participant d'una regata *race*

Taula 4.17: Cas d'ús Insertar SplitTime arribada race

Usuari	*Sistema*
1.- L'usuari prem el botó Arribada i selecciona d'un llistat el dorsal que ha arribat.	
	2.- El sistema captura l'hora GPS i envia la petició d'arribada d'un participant juntament amb l'id del participant afectat.
3.- L'aplicació mostra la llista de participants indicant quins han acabat la regata i quins encara no.	

Taula 4.18: Curs típic d'esdeveniments Insertar SplitTime arribada race

CAS D'ÚS	**Insertar SplitTime arribada rally**
ACTORS	Usuari PDA
PROPÒSIT	Fer una arribada d'un participant d'una regata *rally*
RESUM	L'usuari enregistra al sistema una arribada d'un participant d'una regata *rally*

Taula 4.19: Cas d'ús Insertar SplitTime arribada rally

Usuari	*Sistema*
1.- L'usuari prem el botó Arribada i escriu manualment el dorsal afectat.	
	2.- El sistema captura l'hora GPS i envia la petició d'arribada d'un participant juntament amb l'id del participant afectat.
3.- L'aplicació mostra la vista de regata tipus *rally.*	

Taula 4.20: Curs típic d'esdeveniments Insertar SplitTime arribada rally

CAS D'ÚS	**Anul·lar SplitTime sortida race**
ACTORS	Usuari PDA
PROPÒSIT	Anul·lar l'inici d'una regata *race*
RESUM	L'usuari anul·la l'inici d'una regata tipus *race*

Taula 4.21: Cas d'ús Anul·lar SplitTime sortida race

Usuari	*Sistema*
1.- L'usuari prem el botó anul·lar sortida de la vista de sortida d'una regata *race*	
	2.- El sistema envia la petició d'anul·lar l'inici de regata.
3.- La regata torna a estar en estat pendent i a punt d'iniciar-se	

Taula 4.22: Curs típic d'esdeveniments Anul·lar SplitTime sortida race

CAS D'ÚS	**Anul·lar SplitTime sortida rally**
ACTORS	Usuari PDA
PROPÒSIT	Anul·lar l'inici d'una regata *rally*
RESUM	L'usuari anul·la l'inici d'una regata tipus *rally*

Taula 4.23: Cas d'ús Anul·lar SplitTime sortida rally

Usuari	*Sistema*
1.- L'usuari selecciona el botó anul·lar inici regata.	
	2.- El sistema envia la petició d'anul·lar inici de regata.
3.- L'aplicació mostra la regata com a pendent un altre cop.	

Taula 4.24: Curs típic d'esdeveniments Anul·lar SplitTime sortida rally

CAS D'ÚS	**Anul·lar SplitTime arribada race**
ACTORS	Usuari PDA
PROPÒSIT	Anul·lar una arribada d'un participant d'una regata *race*
RESUM	L'usuari anul·la una arribada d'un participant d'una regata *race*

Taula 4.25: Cas d'ús Anul·lar SplitTime arribada race

Usuari	*Sistema*
1.- L'usuari selecciona un participant i prem el botó Anul·lar Arribada.	
	2.- El sistema envia la petició d'anulació d'arribada d'un participant juntament amb l'id del participant afectat.
3.- L'aplicació mostra la llista de participants indicant quins han acabat la regata i quins encara no.	

Taula 4.26: Curs típic d'esdeveniments Anul·lar SplitTime arribada race

CAS D'ÚS	**Anul·lar SplitTime arribada rally**
ACTORS	Usuari PDA
PROPÒSIT	Anul·lar una arribada d'un participant d'una regata *rally*
RESUM	L'usuari anul·la al una arribada d'un participant d'una regata *rally*

Taula 4.27: Cas d'ús Anul·lar SplitTime arribada rally

Usuari	*Sistema*
1.- L'usuari prem el botó Anul·lar Arribada i escriu manualment el dorsal afectat.	
	2.- El sistema envia la petició d'anul·lació d'arribada d'un participant juntament amb l'id del participant afectat.
3.- L'aplicació mostra la vista de regata tipus *rally.*	

Taula 4.28: Curs típic d'esdeveniments Anul·lar SplitTime arribada rally

4.1.1.5.- Actualitzacions competició

Diagrama de casos d'ús

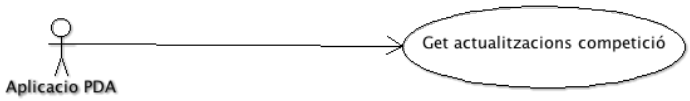


Diagrama 4.5: Diagrama de casos d'ús Actualitzacions competició

Especificació dels casos d'ús

CAS D'ÚS	**Get actualitzacions competició**
ACTORS	Aplicació PDA
PROPÒSIT	Actualitzar l'estat de la competició que s'està tractant.
RESUM	Periòdicament l'aplicació va al servidor a buscar l'estat de la competició.

Taula 4.29: Cas d'ús Get actualitzacions competició

Sistema
1.- Periòdicament, la PDA realitza una crida al servidor demanant l'estat de la competició.
2.- Una vegada el sistema li retorna la informació, l'aplicació la parseja i modifica aquells elements que s'hagin vist afectats per les actualitzacions.

Taula 4.30: Curs típic d'esdeveniments Get actualitzacions competició

4.1.2.- Casos d'ús aplicació pantalla

4.1.2.1.- Seleccionar competició

Diagrama de casos d'ús

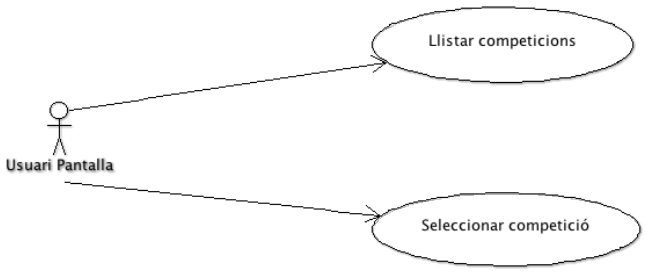


Diagrama 4.6: Diagrama de casos d'ús Seleccionar competició

Especificació dels casos d'ús

CAS D'ÚS	**Llistar competicions**
ACTORS	Usuari Pantalla
PROPÒSIT	Llistar les diferents competicions del sistema
RESUM	A l'usuari se li mostra el llistat de competicions per a poder seleccionar la desitjada

Taula 4.31: Cas d'ús Llistar competicions

Usuari	***Sistema***
1.- L'usuari de la pantalla executa l'aplicació per visualitzar les dades.	
	2.- El sistema executa una crida al servidor i obté totes les competicions disponibles per ser visualitzades.

Taula 4.32: Curs típic d'esdeveniments Llistar competicions

4.1.2.2.- Visualitzar competició

Diagrama de casos d'ús

Diagrama 4.7: Diagrama de casos d'ús Visualitzar competició

Especificació dels casos d'ús

CAS D'ÚS	**Visualitzar competició**
ACTORS	Usuari Pantalla
PROPÒSIT	Visualitzar una competició
RESUM	L'usuari selecciona una competició i aquesta es mostra

Taula 4.33: Cas d'ús Visualitzar competició

Usuari	***Sistema***
1.- L'usuari selecciona una competició del llistat de competicions existent.	
	2.- El sistema realitza una petició al servidor per aconseguir totes les dades de la competició i construeix la interfície gràfica corresponent.

Taula 4.34: Curs típic d'esdeveniments Visualitzar competició

4.1.2.3.- Actualitzacions competició

Diagrama de casos d'ús

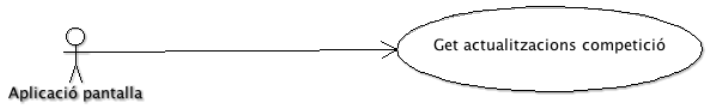


Diagrama 4.8: Diagrama de casos d'ús Actualitzacions competició

Especificació dels casos d'ús

CAS D'ÚS	**Insertar SplitTime**
ACTORS	Aplicació pantalla
PROPÒSIT	Aconseguir les actualitzacions de la competició.
RESUM	Periòdicament l'aplicació aconsegueix l'estat de la competició i l'actualitza a la pantalla.

Taula 4.35: Cas d'ús Get actualitzacions competició

Sistema
1.- Periòdicament el sistema realitza una petició al servidor per obtenir l'estat de la competició
2.- Una vegada té l'estat, actualitza tots aquells elements que ho necessitin i torna a mostrar la interfície gràfica.

Taula 4.36: Curs típic d'esdeveniments Get actualitzacions competició

4.1.3.- Casos d'ús aplicació representació dades

4.1.3.1.- Insertar/Anul·lar SplitTime

Diagrama de casos d'ús

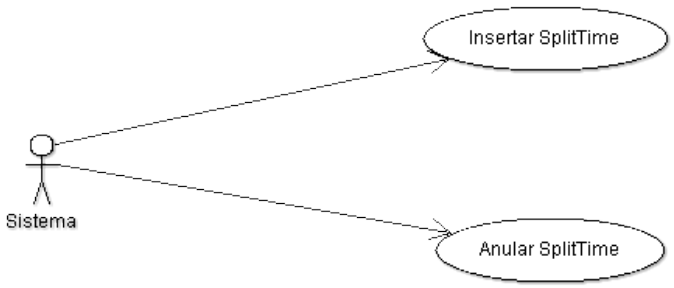


Diagrama 4.9: Diagrama de casos d'ús Insertar/Anul·lar SplitTime

Especificació dels casos d'ús

CAS D'ÚS	**Insertar SplitTime**
ACTORS	Sistema
PROPÒSIT	Plasmar al Spreadsheet la inserció
RESUM	El sistema guarda la informació al Spreadsheet i a la Base de Dades

Taula 4.37: Cas d'ús Insertar SplitTime

Sistema
1.- El sistema rep una petició d'insertar SplitTime amb una competició, rol, regata, dorsal i temps associat.
2.- El sistema genera el domini, guarda la nova informació al lloc corresponent i torna a guardar el domini tant al Spreadsheet com a la Base de Dades.

Taula 4.38: Curs típic d'esdeveniments Insertar SplitTime

CAS D'ÚS	**Anul·lar SplitTime**
ACTORS	Sistema
PROPÒSIT	Plasmar al Spreadsheet l'anulació de temps
RESUM	El sistema anul·la un temps i guarda la informació al Spreadsheet i a la Base de Dades

Taula 4.39: Cas d'ús Anul·lar SplitTime

Sistema
1.- El sistema rep una petició d'anul·lar un temps amb una competició, rol, regata i dorsal associats.
2.- El sistema genera el domini, esborra el temps corresponent i torna a guardar el domini tant al Spreadsheet com a la Base de Dades.

Taula 4.40: Curs típic d'esdeveniments Anul·lar SplitTime

4.1.3.2.- Gestionar errors Spreadsheet

Diagrama de casos d'ús

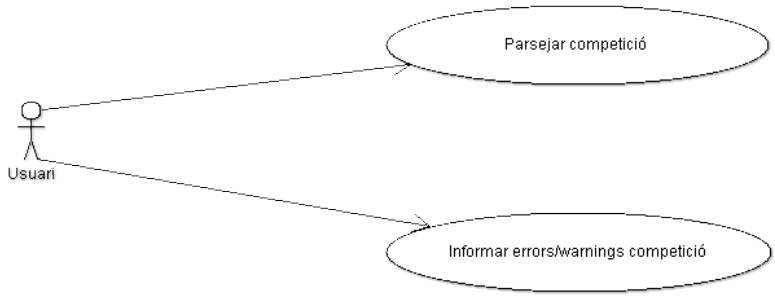


Diagrama 4.10: Diagrama de casos d'ús Gestionar errors Spreadsheet

Especificació dels casos d'ús

CAS D'ÚS	**Parsejar competició**
ACTORS	Usuari sistema
PROPÒSIT	Aconseguir els errors i warnings que hi hagi al Spreadsheet d'una competició
RESUM	L'usuari, un cop ha generat manualment la competició, comprova els errors i warnings que hi pugui haver.

Taula 4.41: Cas d'ús Parsejar competició

Usuari	***Sistema***
1.- L'usuari, un cop ha insertat tota la informació de la competició, clica el botó Generar errors competició per comprovar si hi ha algun error.	
	2.- El sistema extreu la informació del Spreadsheet i comprova tots els possibles errors i warnings que hi pugui haver i els guarda en una llista. El sistema també avisa l'usuari que s'estan processant els errors i warnings.

Taula 4.42: Curs típic d'esdeveniments Parsejar competició

CAS D'ÚS	**Informar errors/warnings competició**
ACTORS	Usuari sistema
PROPÒSIT	Mostrar els errors i warnings que hi hagi al Spreadsheet d'una competició
RESUM	El sistema mostra a l'usuari els errors i warnings que hi ha la competició que ha generat.

Taula 4.43: Cas d'ús Informar errors/warnings competició

Usuari	*Sistema*
	1.- El sistema escriu al final de cada línia afectada, l'error o warning corresponent amb una explicació de l'errada comesa. A més, també mostra un resum de quants errors i quants warnings existeixen.
2.- L'usuari pot corregir els errors i tornar a parsejar la competició per veure que tot estigui correcte.	

Taula 4.44: Curs típic d'esdeveniments Parsejar competició

4.1.3.3.- Actualitzacions Base de Dades

Diagrama de casos d'ús

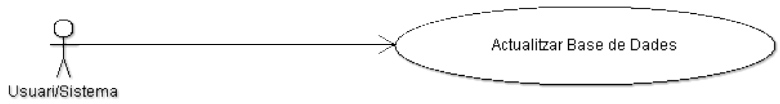


Diagrama 4.11: Diagrama de casos d'ús Actualitzacions Base de Dades

Especificació dels casos d'ús

CAS D'ÚS	**Actualitzar Base de Dades**
ACTORS	Usuari/Sistema
PROPÒSIT	Mantenir la Base de Dades actualitzada
RESUM	El sistema agafa la informació que hi ha al Spreadsheet i actualitza la Base de Dades.

Taula 4.45: Cas d'ús Actualitzar Base de Dades

Usuari	*Sistema*
1.- L'usuari clica el botó Generar Base de Dades.	1.- Periòdicament el sistema crida a actualitzar Base de Dades.
	2.- El sistema genera la informació corresponent a la competició especificada i la guarda a la Base de Dades.

Taula 4.46: Curs típic d'esdeveniments Actualitzar Base de Dades

4.1.3.4.- Generar competició

Diagrama de casos d'ús

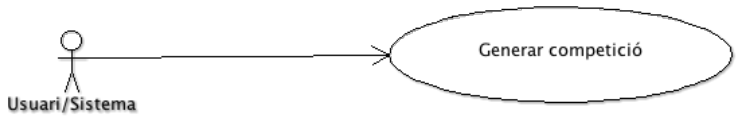


Diagrama 4.12: Diagrama de casos d'ús Generar competició

Especificació dels casos d'ús

CAS D'ÚS	**Generar competició**
ACTORS	Sistema
PROPÒSIT	Extreure la informació que hi ha al Spreadsheet i generar un domini a partir d'aquesta.
RESUM	El sistema transforma les dades del Spreadsheet en un domini per poder interactuar amb ell.

Taula 4.47: Cas d'ús Generar competició

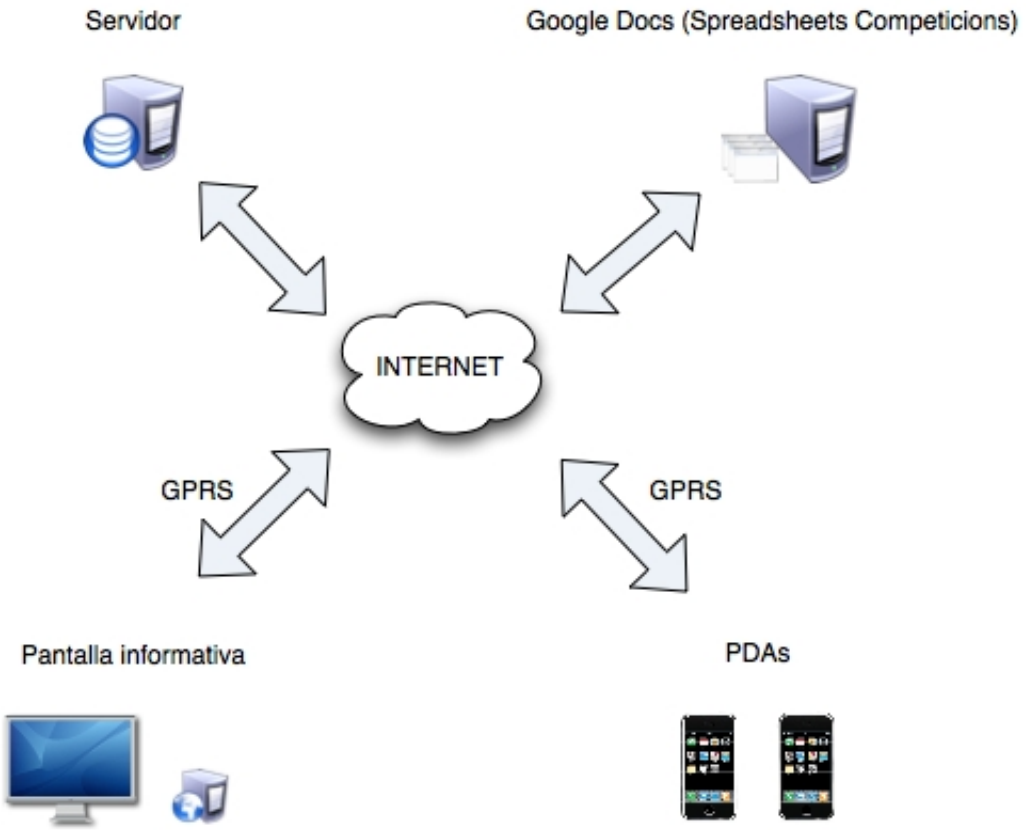
Sistema
1.- El sistema carrega tota la informació existent al Spreadsheet en memòria.
2.- El sistema transforma aquesta informació en un domini sobre el qual poder raonar mitjançant codi.

Taula 4.48: Curs típic d'esdeveniments Generar competició

5.-DISSENY

5.1.-Arquitectura del sistema

5.1.1.- Arquitectura



Imatge 5.1: Arquitectura del sistema

La imatge 5.1 mostra un esquema representatiu de com està muntat el sistema. És fàcil veure que tots els elements que formen el sistema es connecten entre ells a través d'Internet. Això suposa que, per al bon funcionament del sistema, s'haurà de disposar sempre de connexió a la xarxa. A més a més, per a dos dels elements del diagrama, és essencial també disposar de connexió GPRS ja que es trobaran en llocs físics on no sol haver-hi connexió wifi (rius, llacs,...).

A més, es poden observar clarament les 4 subparts en les quals se subdivideix el sistema. A continuació es descriuen mínimament aquestes 4 parts. No obstant, posteriorment es dedicarà un subapartat per a cadascuna d'elles.

El **servidor de GoogleDocs,** on estan emmagatzemats els Spreadsheets. És un servei que ofereix Google i que és gratuït. Aquest servidor emmagatzema els fulls de càlcul que representen les competicions. L'usuari encarregat de crear les competicions, accedeix a aquest servidor mitjançant l'aplicació web Google Docs. A més, en cas de necessitat es poden descarregar les dades de manera local.

La **pantalla informativa**, que es trobarà situada al lloc on s'està realitzant la competició i que mostrarà al públic com transcorre la competició. Aquesta aplicació es connectarà periòdicament al servidor per descarregar-se l'estat actual de la competició.

Les **PDAs**, que s'encarregaran de generar els temps que s'aniran guardant al Spreadsheet de la competició. S'haurà de pensar també com es comuniquen la PDA de sortida i la d'arribada per assegurar que el crono de cadascuna sigui exactament el mateix.

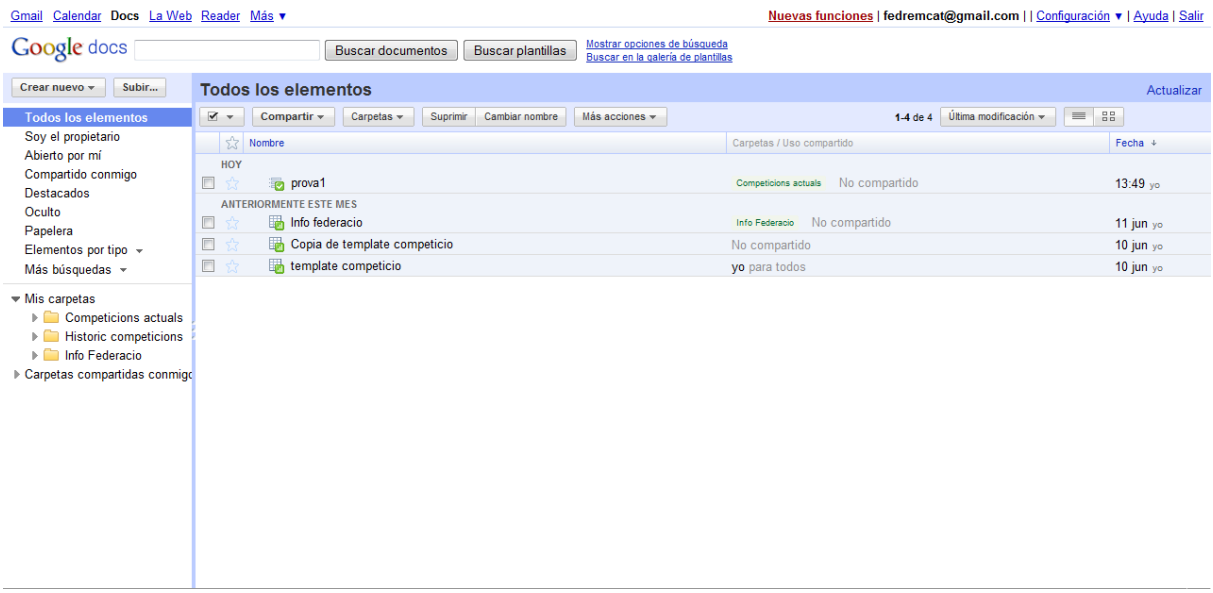
El **Servidor**, que tindrà diverses funcionalitats. Per una banda atendre les peticions que vinguin tant de l'aplicació de la pantalla com de les PDAs. Aquestes peticions poden ser: inserir un temps de sortida, inserir un temps d'arribada, generar l'estat de la competició en un moment donat, etc. Per altra banda aquest servidor és l'encarregat d'hostatjar la web de consulta de l'històric de competicions.

5.1.2.- Servidor GoogleDocs

És un servidor de Google en el qual estan penjats els Spreadsheets que representen les competicions que es van generant. Aquí es guarden tant les competicions actuals com l'històric de competicions. A més, també es guarda un Spreadsheet molt important que conté tota la informació de la federació en quant a clubs, persones i modalitats.

Un avantatge que proporciona aquesta manera de representar les dades és que aquestes estan sempre disponibles i que són accessibles des de qualsevol punt amb connexió a Internet. Per tant, és com si portéssim sempre les dades a sobre, però amb el gran avantatge de que no ens les podem oblidar enlloc ja que, en realitat, no les portem.

El principal avantatge d'utilitzar aquest sistema però, és que Google Docs ofereix un gran servei que permet la modificació remota de les dades tan per part d'un humà com a través de codi. Més encara, quan quelcom o algú modifica informació a algun document, aquesta es fa persistent al moment. De manera que si el sistema rep dues peticions de modificació d'informació en un curt període de temps, la segona petició que s'atengui ja veurà la informació tal i com l'ha deixat la modificació feta per la primera petició.



Imatge 5.2: Captura pantalla GoogleDocs

La imatge 5.2 mostra com és la interfície que ofereix GoogleDocs. Aquesta aplicació funciona de manera molt similar a un gestor de correu electrònic. Es poden crear nous elements, modificar-ne de ja existent, esborrar-ne, compartir-los amb altres usuaris i un llarg etcètera de funcionalitats.

Una característica que ens és útil per a Iris és la gestió que es fa dels elements. M'explico, com es pot observar a la part esquerra de la imatge 5.2, Google ofereix la possibilitat de crear carpetes. A més, mitjançant codi, es pot realitzar algun tipus d'acció als elements que es trobin a una carpeta concreta. Això és de vital importància per a Iris ja que d'aquesta manera es podran separar les competicions ja finalitzades i que no reben canvis de les actuals, que estan contínuament modificant-se. Quan una competició s'ha acabat és tan simple com arrossegar-la fora de la carpeta de competicions actuals perquè deixi de comportar-se com a tal.

Una altra característica que demostra la potència d'aquesta aplicació és que permet crear plantilles, encara que aquí els anomenarem *templates*[12]. Els templates són de gran utilitat per a l'usuari ja que no haurà de crear una competició des de zero cada vegada, sinó que la crearà des del template. Així l'usuari disposarà d'un llistat de regates d'exemple que utilitzant copiar i apegar podrà anar fent més i més gran. Aquesta metodologia li suposarà un esforç molt més reduït que si hagués d'escriure tota la informació a mà.

5.1.3.- PDAs

Les PDAs són els elements del sistema que canvien l'estat de les competicions. Més concretament modifiquen les dades de les regates i participacions d'aquestes competicions a petició dels usuaris d'aquestes PDAs.

Han d'existir com a mínim dues PDAs, una que tindrà el rol d'àrbitre de sortida i l'altra que farà d'àrbitre d'arribada. Clarament la PDA de sortida serà l'encarregada d'iniciar les regata al sistema, mentre que la PDA d'arribada serà l'encarregada d'anar inserint els temps d'arribada de cada participant de la regata.

Per solucionar el problema que s'esmentava a l'anàlisi de requeriments sobre la de-sincronització entre àrbitres, s'ha pensat la següent solució:

Per una banda ens trobem amb el problema de representació de temps. Cada PDA té el seu propi rellotge intern, el qual no té per què coincidir (i de fet mai és exactament igual) entre PDA. Aquesta limitació va fer pensar en una representació universal del temps. I la millor solució trobada va ser utilitzar el receptor GPS que porten incorporades les PDAs. D'aquesta manera el temps és el mateix independentment del hardware en el qual ens trobem.

Per altra banda, per solucionar el problema de de-sincronització es necessita comunicació entre les PDAs de sortida i les d'arribada. Amb aquesta comunicació es pot enviar el moment en que s'ha iniciat la regata (amb hora GPS) de la PDA de sortida cap a la PDA d'arribada. D'aquesta manera desapareix totalment el problema.

Degut a la desconeixença que es tenia a l'empresa amb aquest tipus de hardware, s'han realitzat diverses proves per veure quina és la millor manera de comunicar dues PDAs. Les proves que s'han realitzat són:

1.- Comunicació directa entre PDAs mitjançant sockets[13]: El resultat d'aquesta prova no va ser productiu ja que era excessivament lent i en ocasions fins i tot es tallava la connexió entre les PDAs (per política de la companyia telefònica que dóna connexió GPRS a les PDAs).

2.- Comunicació PDA sortida – Servidor i Servidor – PDA Arribada mitjançant sockets: Muntant el sistema d'aquesta manera es van tenir els mateixos problemes de lentitud i talls de connexió que al punt 1.

3.- Comunicació PDA sortida – Servidor i Servidor – PDA Arribada mitjançant crides a Web Services: Veient que potser el problema eren els sockets, es va provar aquest altre mètode. Els resultats obtinguts van ser satisfactoris, i els problemes que plantejaven els dos altres punts van desaparèixer. Per tant es va decidir utilitzar aquesta tecnologia.

5.1.4.- Pantalla informativa

És la pantalla que mostrarà l'estat d'una competició en un moment determinat. Aquesta pantalla estarà alimentada per un PC on s'estarà executant una aplicació. Aquesta aplicació s'anirà connectant al servidor cada cert temps (s'ha pensat cada 20 segons aprox.).

Aquesta pantalla està pensada per utilitzar-se al mateix lloc on es desenvolupen les competicions, és a dir, llocs on es fàcil que no hi hagi connexió wifi. Per altra banda, s'ha explicat que tots els elements necessiten connexió a Internet. Per tant necessitarem alguna alternativa a la connexió wifi, i el que s'ha cregut convenient és la utilització d'un mòdem GPRS.

Finalment dir que encara es desconeix com s'implementarà aquesta aplicació. Una de les possibilitats seria utilitzar Java i les seves llibreries d'interfícies gràfiques. Tot i això, aquesta opció està pràcticament descartada ja que aquestes llibreries es troben bastant obsoletes. A més a més, no produeixen uns resultats vistosos que és un dels requeriments principals d'aquesta part del sistema.

Afortunadament existeixen altres solucions per implementar aquesta aplicació com poden ser Flash, JavaFx, Open GL, etc. En el seu moment es decidirà quina d'aquestes opcions es creu que és la més indicada per al que es vol fer.

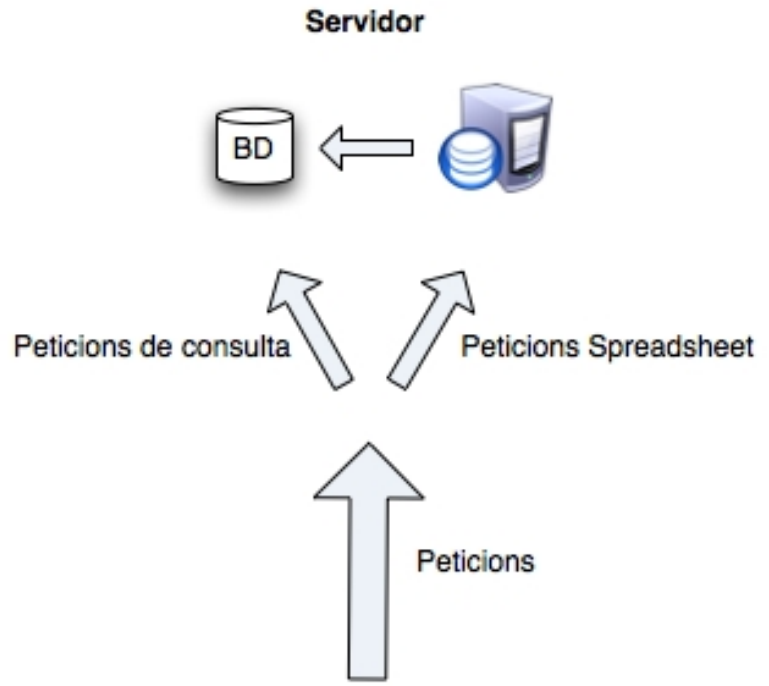
5.1.5.- Servidor

El servidor és l'encarregat de rebre les peticions que es facin al sistema i reaccionar de la manera convenient. En altres paraules, qualsevol crida que es realitzi al sistema, s'envia al servidor i aquest, depenent de la petició, retornarà la informació sol·licitada o delegarà la responsabilitat al responsable d'aquesta.

D'aquesta definició es deriva una problemàtica, la concurrència al servidor. És possible i bastant probable que arribin al servidor dues peticions que volen accedir i/o manipular els mateixos recursos. Per tant, s'haurà de pensar un mecanisme per evitar aquests conflictes. Una bona manera d'evitar-ho és obligar les crides a adquirir el domini abans de realitzar qualsevol tasca i alliberar-lo un cop finalitzada. D'aquesta manera quan una crida hagi adquirit el domini, cap altra tasca podrà adquirir-lo fins que la primera no finalitzi i l'alliberi.

Per altra banda, com ja s'ha dit anteriorment en aquesta memòria, extreure la informació d'un Spreadsheets amb gran volum de dades és bastant costós temporalment parlant. Això sumat al que s'explica a l'apartat anterior pot produir problemes de gran quantitat de peticions esperant a ser ateses (un símil per entendre-ho serien les cues que es generen als peatges).

Després de meditar-ho durant un temps s'ha decidit introduir una Base de Dades a la lògica del sistema.



Imatge 5.3: Diagrama del Servidor afegint la Base de Dades

La imatge 5.3 mostra com quedaria el sistema afegint-li aquesta Base de Dades. El funcionament seria el següent:

El servidor, cada cert període de temps, guardarà a la Base de Dades la rèplica de les competicions que s'estiguin tractant en aquest moment, és a dir, les que es troben a la carpeta *Competicions actuals* de GoogleDocs (veure figura 5.2).

Les peticions que pot rebre el sistema es divideixen en dues parts, les de consulta i les de modificació de dades. Doncs el que s'intenta amb aquesta Base de Dades és evitar que les operacions de consulta hagin d'adquirir el domini.

Ara, quan arribi una petició, si és de consulta d'informació (pantalla o PDAs) es consultarà la Base de Dades sense necessitat d'adquirir el domini. Per contra, quan la petició és de modificació (PDAs) el sistema es comporta tal com ho feia anteriorment, adquirint el domini a l'inici de la transacció i alliberant-lo a l'acabar. A més cal dir que quan arriba una petició d'aquest tipus també es modifica la Base de Dades i amb això es guanya en 2 aspectes:

1.- La informació estarà consultable al moment i no farà falta esperar que el servidor realitzi la còpia periòdica per poder consultar les actualitzacions.

2.- Les modificacions que puguin inserir els usuaris directament al Spreadsheet es veuran reflectides a la Base de Dades, i per tant seran contemplades per les operacions de consulta, en el moment en que el servidor realitzi la còpia periòdica.

En definitiva, amb aquesta nova arquitectura i funcionament del servidor, solucionem el problema d'accés massiu al sistema que comentàvem anteriorment i, a més, aconseguim optimitzar una mica el sistema sacrificant mínimament la disponibilitat temporal de les dades.

5.2.- Disseny aplicació pantalla

Una vegada definits els requeriments i l'arquitectura que ha de tenir l'aplicació de la pantalla és moment de dissenyar com haurà de ser l'aplicació que es comuniqui amb el servidor i mostri les dades per la pantalla.

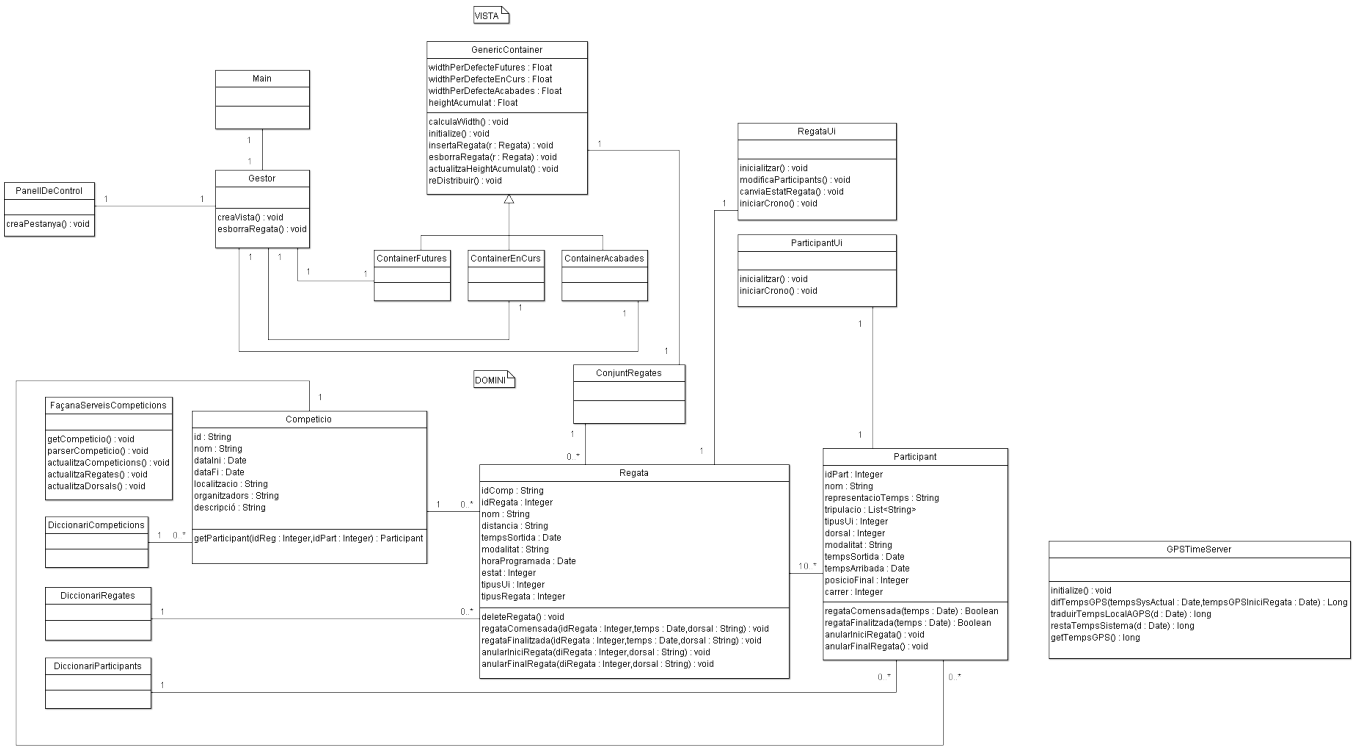


Diagrama 5.1: UML Pantalla

El diagrama 5.1 mostra l'UML dissenyat per a l'aplicació de la pantalla. Donat que possiblement es fa difícil de llegir, a l'annex A hi ha el mateix diagrama amb un format més gran i més fàcil de llegir.

Com es pot observar s'ha dividit el diagrama en capes, en dues capes exactament, el Domini i les Vistes. No hi ha capa de Dades ja que aquesta aplicació no guarda ni modifica dades sinó que aquestes li arriben de forma remota.

Aquesta divisió del diagrama ve d'aplicar el patró model-vista-controlador[14] (MVC). Aquest patró intenta dividir el sistema en una part de vistes, una part de domini i una sèrie de controladors que s'encarreguen de modificar l'estat de les vistes i/o del domini.

5.2.1.- Domini

El domini està dividit en dues parts principals. Per una banda tenim la part del model que representa les competicions. Com es pot observar una competició està formada per un conjunt de regates. Al mateix temps, una regata està formada per un conjunt de participants. Quan es reben les dades d'una competició, aquestes es parsegen i es transformen en instàncies d'aquestes classes.

Per altra banda tenim els diccionaris de competicions, regates i participants. Aquests diccionaris mantenen un registre de quines competicions, regates i participants hi ha actius al sistema. Així, quan arriba una actualització amb noves dades, s'actualitzen aquests diccionaris i s'esborren del sistema totes aquelles dades que no s'han rebut amb la última actualització.

En tercer lloc tenim la Façana de serveis. La responsabilitat d'aquesta façana és realitzar les crides al servidor per aconseguir les dades. Una vegada aconseguides les dades, les parseja i crea (o actualitza si ja existien) les instàncies pertinents de competició, regata i participant. A més realitza l'actualització dels diccionaris que s'esmenta a l'anterior paràgraf.

Finalment tenim el servidor de temps GPS. Aquesta classe té la finalitat de comunicar-se amb el receptor GPS i manipular les trames que aquest rep per aconseguir el temps GPS actual. Aquest temps s'utilitzarà per mostrar a vistes el cronòmetre de cadascuna de les regates que hi hagi en curs.

5.2.2.- Vistes

A la part de vistes hi tenim diversos elements. El punt de partida és la classe Main. Aquesta classe s'encarregarà de crear l'escenari on es mostraran les regates. A més, també té la responsabilitat d'inicialitzar la classe gestor.

Abans de parlar del gestor, però explicarem el concepte de Contenidor. Un contenidor és un element que es pot inserir a la imatge per ser visualitzat i que, a més, pot (com el seu nom indica) contenir altres elements. En el nostre cas aquests altres elements seran instàncies de la classe RegataUI i a la vegada, cada RegataUI tindrà un contenidor on es mostraran les instàncies de ParticipantUI corresponents a aquesta regata. Com es pot suposar les classes RegataUI i ParticipantUI són les representacions gràfiques de les classes Regata i Participant definides a domini.

La nostra aplicació tindrà 3 contenidors de RegatesUI (amb una llista de regates associada a cadascun) mostrant-se en tot moment per pantalla. Aquests són:

- Contenidor regates futures: conté les regates que estan pendents d'iniciar-se.
- Contenidor regates en curs: conté les regates que estan en curs en un moment donat.
- Contenidor regates finalitzades: conté les regates que ja s'han finalitzat.

S'ha previst que aquests contenidors, en algun cas, poden contenir més regates de les que es poden mostrar en pantalla en un moment donat. Per tant, s'ha afegit lògica per anar refrescant la pantalla i assegurar que totes les regates es van mostrant durant un temps.

El gestor és qui porta el control del què s'està mostrant per pantalla en un moment determinat. Per una banda, controla el contingut dels 3 contenidors esmentats anteriorment. El que fa és controlar per exemple la grandària dels contenidors. En el moment en què un contenidor es passa de la grandària que té definida a priori, se li apliquen les transformacions pertinents. Per altra banda, també té associat una instància de panell de control i reacciona als canvis que es poden inserir a través d'aquest panell.

El panell de control ofereix serveis per gestionar els elements que es mostren a la pantalla. Més concretament permet especificar la grandària que ha de tenir a la pantalla cadascun dels contenidors, el nombre d'elements que es volen mostrar a cada contenidor i com es volen mostrar aquests (en una columna, en dos columnes,...) i el temps de refrescat dels contenidors en cas que aquests no puguin mostrar al mateix temps tots els elements que contenen.

Finalment explicar que la manera de procedir que s'ha pensat utilitzar és:

1.- Es reben les dades i es parsegen per a crear, modificar o esborrar els elements del domini (competicions, regates i participants).

2.- Els elements del domini d'alguna manera (això dependrà de la tecnologia que finalment s'utilitzi per implementar aquesta part) avisaran a les vistes dels canvis que s'han realitzat.

3.- Les vistes modificaran els elements que s'estan visualitzant amb la finalitat de mostrar les dades actualitzades que s'acaben de rebre.

5.3.- Disseny aplicació PDAs

L'aplicació que es realitzarà per a les PDAs mostra el disseny següent:

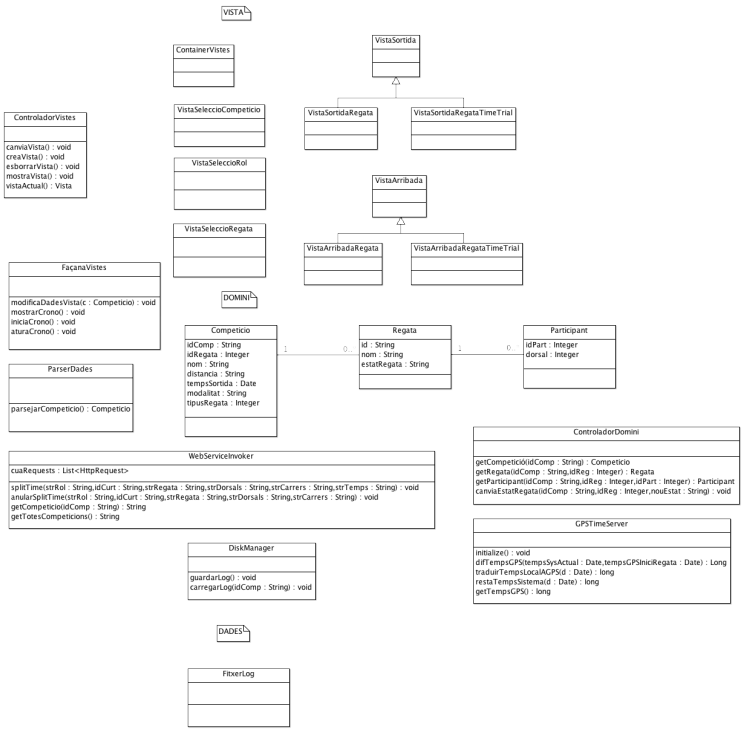


Diagrama 5.2: UML PDAs

El diagrama 5.2 és l'UML resultant del disseny de l'aplicació que utilitzaran les PDAs. Donat que possiblement es fa difícil de llegir, a l'annex A hi ha el mateix diagrama amb un format més gran i més fàcil de llegir.

Com s'hauria de poder observar el diagrama està dividit en tres capes: Vistes, Domini i una petita part de Dades. Per altra banda, existeixen també uns controladors de Vistes i Domini. És fàcil veure doncs que estem utilitzant de nou un patró MVC.

Cal dir que s'ha decidit utilitzar aquest patró als dos llocs per les característiques comunes que presenten les dues aplicacions:

- Les dues tenen una part d'interfície gràfica que han de mostrar als cronometradors de proves en un cas i al públic en l'altre.

- Les dues disposen del mateix model de dades a mostrar a través de les vistes. El model de Competició, Regata i Participant.

- Les dues compten amb controladors que interactuen amb les vistes i les dades.

5.3.1.- Dades

La capa de dades és la més simple de totes. Simplement conté una classe que representa els fitxers de log del sistema. En aquests fitxers es va guardant tota la informació de la competició per tenir un registre d'errors i, possiblement un sistema de recuperació de dades. Això últim és una idea que s'ha tingut però queda pendent decidir si posar-se en pràctica o no.

Els fitxers de log estan gestionats per la classe DiskManager que serà l'encarregada d'escriure i llegir les dades que s'aniran guardant.

5.3.2.- Domini

El domini, que té una certa semblança al de l'aplicació anterior, consta de diverses parts. Per una banda tenim el model de dades que es pràcticament idèntic al de la pantalla i està format per competicions, regates i participants.

En segon lloc tenim els mecanismes de connexió amb el servidor que ens permetran l'intercanvi de dades amb aquest. La classe encarregada d'aquesta part és WebServiceInvoker on s'ofereixen els serveis d'enviar o anul·lar un SplitTime i de rebre les dades d'una competició o fins i tot de rebre les dades de totes les competicions. Cal dir que aquests serveis estan detallats més endavant a la part on s'explica el disseny del servidor (veure punt 5.4.4).

Addicionalment, aquesta classe conté una cua en la qual s'aniran insertant les peticions que es van realitzant. El primer element d'aquesta cua serà el següent en realitzar la crida, i així successivament. Si alguna d'aquestes peticions no és atesa o ha retornat error, es tornarà a encuar, d'aquesta manera no es perden peticions.

També tenim la classe ControladorDomini, un controlador que s'encarregarà de generar el model de dades i interactuar amb aquest. Una de les funcions que realitza és que cada cert temps rebrà una actualització procedent del servidor i haurà de modificar el model de dades.

Finalment i igual que a l'apartat anterior, tenim el servidor de temps GPS. Aquesta classe té la finalitat de comunicar-se amb el receptor GPS (integrat a la PDA en aquest cas) i aconseguir el temps GPS actual. Aquest temps s'utilitzarà per mostrar a les PDAs els cronòmetres pertinents.

5.3.3.- Vistes

Pel que fa a les vistes el diagrama conté una gran quantitat d'elements a mostrar. Les vistes es podrien dividir en vistes de definició i vistes d'acció. Tot seguit es classifiquen aquestes vistes i es fa una breu explicació del seu contingut.

5.3.3.1.- Vistes de definició

Són aquell conjunt de vistes que ajuden al sistema a definir quin rol tenim i si anem a interactuar amb una regata de tipus *race* o de tipus *rally*.

Abans de començar però, és bo definir quins rols existeixen en aquesta aplicació. Per un cantó tenim el rol d'àrbitre de sortida, aquest estarà posicionat a la sortida i s'encarregarà d'iniciar les regates. Per l'altre cantó tenim el rol d'àrbitre d'arribada, aquest està posicionat a l'arribada i va aturant el crono dels diferents participants.

A més a més també s'hauria d'explicar que existeixen dos tipus de regates. Unes són les regates tipus *race* en les quals tots els participants tenen exactament el mateix temps de sortida i van fent arribades en diferents moments de temps. I les altres, les regates tipus *rally* on cada participant té un temps de sortida diferent.

Una vegada clars els conceptes és moment d'explicar quines vistes pertanyen a aquesta part de definició:

- VistaSeleccioCompeticio: És la vista que ens permet seleccionar la competició desitjada d'entre totes les possibles.

- VistaSeleccioRol: Clarament és la vista on escollim si volem ser àrbitre de sortida o d'arribada.

- VistaSeleccioRegata: En aquesta vista es mostren el llistat de totes les regates de la competició. Al seleccionar-ne una depenent de si aquesta està definida de tipus *race* o *rally* s'obrirà un tipus de finestra o l'altra.

5.3.3.2.- Vistes de modificació

Són les vistes que modifiquen les dades i inicien les peticions de crida al servidor per fer enviar-li la informació. Aquestes vistes són:

- VistaSortidaRegata: És la vista per realitzar la sortida de les regates tipus *race*. Donat que tots els participants han de tenir exactament el mateix temps de sortida, aquesta vista només mostra un botó per iniciar-se i un altre per anul·lar la sortida.

- VistaSortidaRegataTimeTrial: És la vista per iniciar regates tipus *rally*. Com cada participant sortirà en un temps diferent, necessitem poder iniciar en aquesta vista els participants un a un.

- VistaArribadaRegata/VistaArribadaTimeTrial: Aquestes vistes realitzen arribades dels participants. Per tant, també necessitarem realitzar les arribades una per una informant de quin participant ha arribat. En principi no es veu cap diferència entre aquestes dues vistes, però s'ha cregut convenient realitzar l'especialització per si sorgeixen canvis.

Finalment tenim dues classes més que ens ajuden a gestionar el sistema. Per una banda tenim la FaçanaVistes que recull totes les peticions de domini cap a vistes. Per altra banda tenim el controlador de vistes que s'encarrega de crear les vistes, modificar-les i esborrar-les i de mostrar en cada moment la vista corresponent.

5.4.- Disseny servidor

Tot seguit es mostra el disseny que s'ha realitzat de la part de gestió de dades. Aquesta part serà l'encarregada de rebre totes les peticions del sistema i realitzar les accions necessàries per a cadascuna.

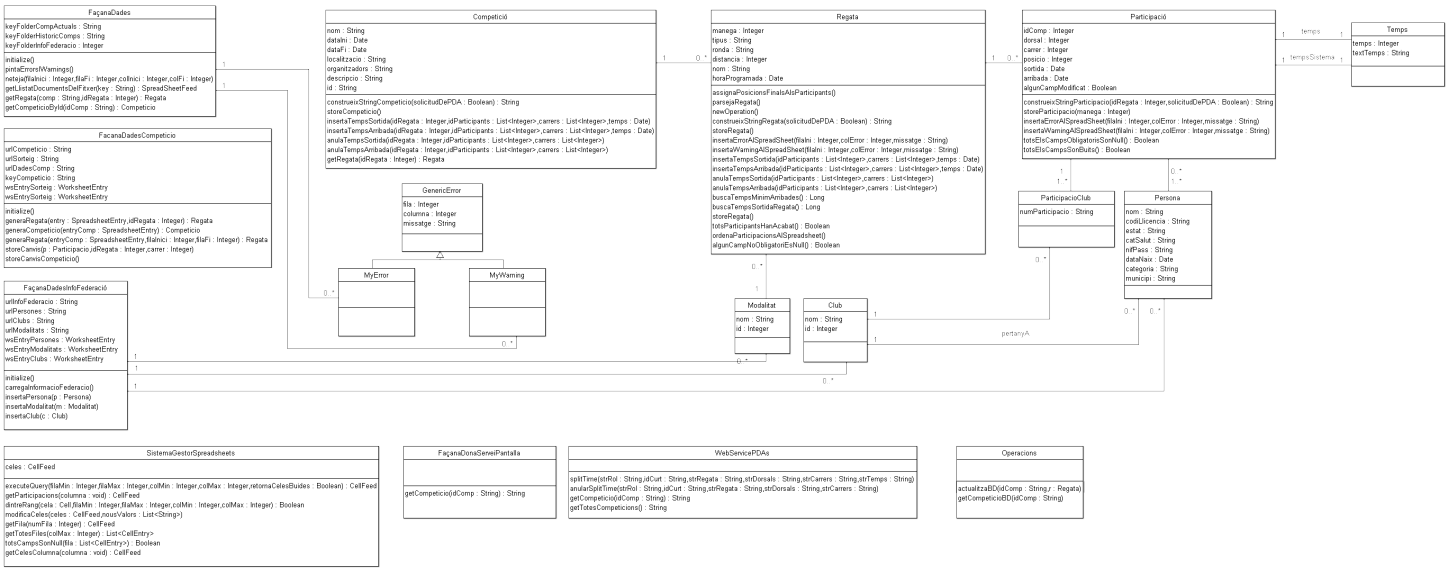


Diagrama 5.3: UML Dades

La figura 5.3 mostra el diagrama UML de la part de gestió de dades. Donat que possiblement es fa difícil de llegir, a l'annex A hi ha el mateix diagrama amb un format més gran i més fàcil de llegir.

Com passava a la part de domini del punt 5.3 el sistema està dividit en dues parts. Primerament tenim el model de dades, que és una abstracció d'una competició amb tots els elements que la conformen (regates, participants, persones, clubs,...). En segon lloc tenim els mecanismes de comunicació amb les diverses parts de l'aplicació. Aquests mecanismes són la comunicació amb els Spreadsheets, amb la pantalla i amb les PDAs. A continuació es detallaran aquestes diverses parts amb les quals està dividit el sistema.

5.4.1.- Model de dades

El model de dades és una abstracció del que és a la vida real una competició. Una competició té un identificador i un conjunt de regates. Aquestes regates s'identifiquen per un identificador de regata més l'identificador de la competició a la que pertany. Les regates són d'una modalitat concreta i tenen un conjunt de participacions associades.

Per la seva banda, una participació s'identifica amb un id (el dorsal de la participació) i els identificadors de competició i regata que la contenen. Les participacions consten d'un llistat de persones que són els remers que formen l'embarcació, i dos temps: el temps de sistema que sempre representa la diferència entre temps d'arribada i de sortida i el temps pròpiament dit, el qual pot contenir un text que s'hagi inserit manualment.

Una participació també té un llistat de participacions de clubs. Aquesta classe permet que un club pugui tenir diverses participacions en una mateixa regata, ja que aquesta situació es pot donar a la vida real.

Finalment, una instància de la classe Persona pertany a un Club i té realitzades fins al moment un conjunt de participacions.

Algú podria pensar que la relació entre Participació i Club (que genera la classe associativa ParticipacioClub) és innecessària ja que aquesta mateixa informació es pot aconseguir navegant des d'una participació a les persones que la conformen i de persona al club al que pertany. Tot i això, a la vida real de les competicions, es permet que una persona participi en una regata remant per a un altre club que no sigui el seu. D'aquesta manera, si no disposéssim de la relació esmentada, es perdria aquesta informació.

5.4.2.- Mecanismes de comunicació amb els Spreadsheets

Existeixen diverses classes encarregades de la comunicació amb els Spreadsheets ja sigui per a consultar, afegir o esborrar informació. Aquestes comunicacions poden anar dirigides a una competició concreta o a informació general de la federació (llistat de persones, de clubs o de modalitats).

La classe FaçanaDades és l'encarregada de gestionar totes les peticions que es reben en relació als Spreadsheets i delegar la responsabilitat a la Façana corresponent, ja sigui la de informació de la federació o la de dades de competició. Una altra de les responsabilitats d'aquesta classe serà plasmar la informació als Spreadsheets, ja sigui informació dels participants (els temps, posicions,...) com informació d'errors i warnings trobats al parsejar la competició. Aquests errors i warnings es guarden en dues llistes que manté aquesta classe i que s'enviaran al Spreadsheet quan l'usuari ho demani.

Les peticions que arriben a la FaçanaDadesCompeticio seran del tipus generar competició, generar regata o guardar canvis. Les operacions de generació de dades agafen una part (o tot) del Spreadsheet i el parsegen per transformar-lo al model corresponent.

Per la seva banda, les peticions que es deleguen a la FaçanaDadesInfoFederació seran de l'estil genera conjunt persones, modalitats i clubs o insertar conjunt persones, modalitats i clubs. És a dir aquesta classe serà l'encarregada de generar totes les instàncies de Persona, Modalitat i Club que existiran al sistema A més també serà la responsable d'inserir noves persones, clubs o modalitats al Spreadsheet d'informació de la federació.

Aquestes classes esmentades tenen el suport de la classe SistemaGestorSpreadsheets. Aquesta classe ofereix serveis per, donat un conjunt de cel·les, poder fer consultes per extreure'n informació. Per exemple, quan s'està construint una competició, s'utilitza aquesta classe per determinar quins continguts de quines cel·les han de ser tractats i quin valor semàntic tenen. El mateix passa quan es vol transformar un conjunt de cel·les a un llistat de Persona o de Modalitat, etc.

5.4.3.- Mecanismes de comunicació amb la pantalla

Aquesta part és la que dóna servei a l'aplicació de la pantalla quan aquesta demana actualitzacions de les dades. Com s'ha dit a l'apartat 5.1.5 les peticions de consulta que es rebin no es delegaran cap al gestor de Spreadsheets, sinó que es consultarà la Base de Dades que es va actualitzant periòdicament.

Així, quan es rebi una petició de pantalla (que portarà un identificador de competició associat), el sistema reaccionarà anant a buscar a la Base de Dades la informació de la competició especificada. Aquesta informació es parsejarà, és a dir, es transformarà al format que espera la pantalla i es guardarà a un String que serà enviat com a resposta a la petició.

5.4.4.- Mecanismes de comunicació amb les PDAs

Finalment tenim els mecanismes de comunicació amb les PDAs. Aquesta part serà un WebService representat en aquest diagrama per la classe WebServicePDAs. Cal dir que existeixen mètodes per transformar una classe Java a un WebService on els serveis que oferirà seran cadascun dels mètodes públics que tingui la classe.

Aquest WebService oferirà els següents serveis:

- **SplitTime/Anular SplitTime:** Aquest servei serà cridat per les PDAs quan vulguin inserir un temps ja sigui d'arribada o de sortida a un participant o conjunt de participants. Clarament l'aplicació inserirà el nou temps i delegarà a la FaçanaDades la responsabilitat de fer persistents les dades al Spreadsheet. A més no ens hem d'oblidar que aquestes crides també modifiquen directament la Base de Dades, així no fa falta esperar a que el servidor la refresqui per disposar de les noves dades. Aquesta tasca es delega a la classe Operacions que és l'encarregada d'interactuar amb la Base de Dades.

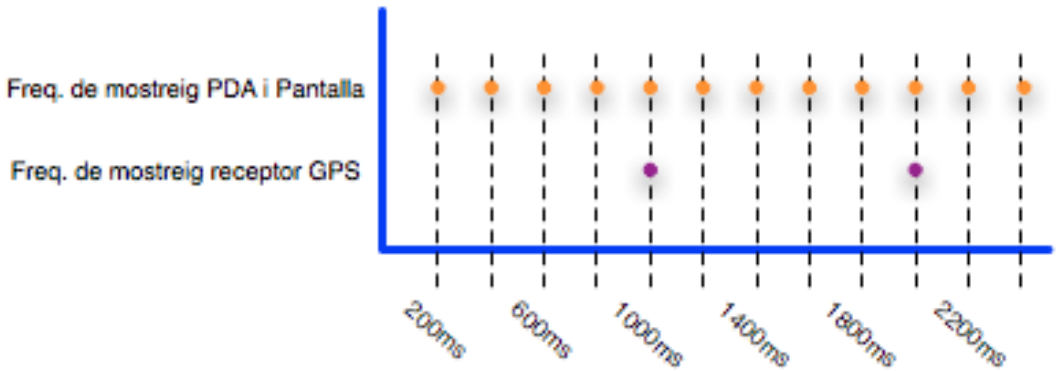
- **Get Competició:** Aquest servei serà cridat des de les PDAs quan per aconseguir les actualitzacions de la competició que se li esta passant per paràmetre. Un altre cop aquestes dades s'extreuen de la Base de Dades i no directament del Spreadsheet.

- **Get totes Competicions:** Com que a priori les PDAs no saben amb quina competició han de treballar, existeix aquesta crida que retorna el llistat de totes les competicions del sistema que es troben a la carpeta *Competicions Actuals* de GoogleDocs. Després d'això l'usuari de la PDA ja seleccionarà la competició que li interessi i les successives crides per aconseguir les actualitzacions ja es realitzaran amb el servei anterior que només retornarà la informació de la regata especificada.

5.5.- Altres consideracions de disseny

Durant el procés de disseny, quan es pensava en com serien les vistes de la pantalla i de les PDAs, es va decidir que s'haurien de posar cronòmetres tant a una aplicació com a l'altra. Es va parlar també de cada quan s'haurien de refrescar aquests cronòmetres. Analitzant les característiques que havien de tenir els cronòmetres es va veure que haurien de contemplar com a mínim fins a les dècimes ja que sinó, no donarien sensació de cronòmetre sinó de rellotge. Per tant, com que hauran d'haver diversos cronos i pensant en no sobrecarregar el sistema, un temps de refrescat de 200ms es va considerar idoni.

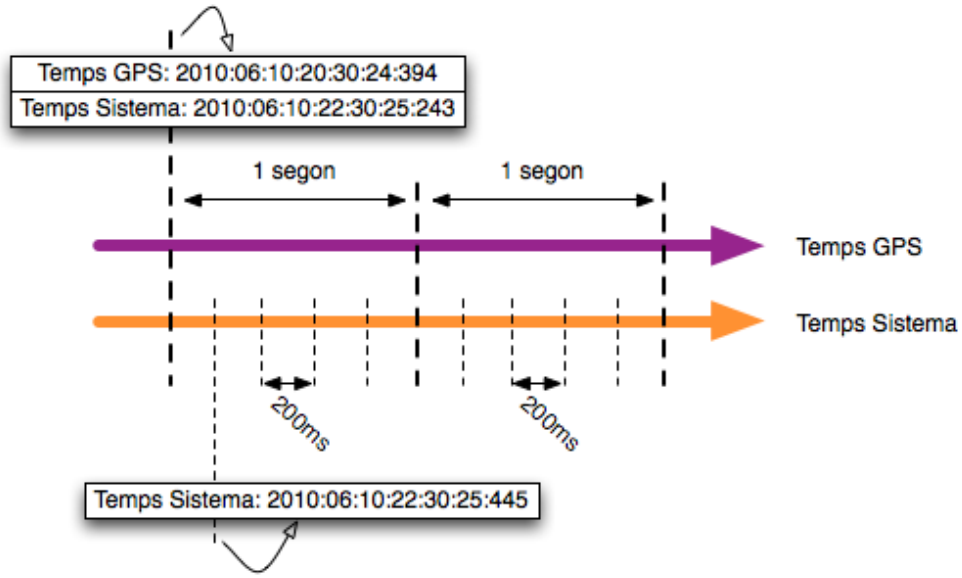
Fins aquí tot està bé, el problema ve quan ens adonem que els receptors GPS (necessaris per al funcionament del sistema) només reben l'hora una vegada per segon. Aquí és on hi ha la inconsistència: no es poden alimentar els cronos amb temps GPS cada 200ms si el receptor GPS només rep dades cada segon.



Imatge 5.4: Comparativa de freqüències d'actualització GPS vs PDA i Pantalla

La imatge 5.4 mostra la comparativa de períodes de mostreig entre els receptors GPS i les PDAs i la Pantalla. És clar que no pot funcionar el sistema amb aquestes premisses ja que el crono només s'incrementaria a ritme de segons.

Afortunadament s'ha trobat una solució al problema. Aquesta solució es basa en aprofitar el temps de sistema del hardware utilitzat, és a dir, de les PDAs i del Pc que alimenta la pantalla com a auxiliar al temps de GPS.



Imatge 5.5: Sistema per calcular el temps

La imatge 5.5 mostra un esquema explicatiu d'aquesta solució. El que es fa és capturar una parella de temps GPS-Sistema cada vegada que el receptor GPS rep el temps, és a dir, cada segon. A partir d'aquí es captura el temps de sistema cada 200ms i s'aplica la següent fórmula:

$$TempsGPS_{Actual} = TempsGPS_{ÚltimaParella} + (TempsSistema_{Actual} - TempsSistema_{ÚltimaParella})$$

És fàcil veure que el que està fent aquesta fórmula és calcular el temps que ha passat des de la rebuda de l'últim temps GPS fins al moment i sumar-lo al temps GPS que es té.

En definitiva, s'ha aconseguit un sistema que emula el funcionament d'un receptor GPS amb període de refresc definible per l'usuari. Més encara del que necessitàvem, ja que a la nostra aplicació, com ja hem dit, n'hi havia prou amb una freqüència de 200ms

Cal dir també que quan es programa un dispositiu per a que reaccioni cada cert temps, com es el cas dels cronòmetres, aquest temps no té perquè ser exacte al 100%, ja sigui per saturació del sistema o altres causes diverses. És molt probable, per tant, que transcorregut un temps, haguem perdut certa precisió. No obstant, com que la nostra aplicació s'actualitza amb temps GPS una vegada per segon, aquest error no es prolongarà més d'això, un segon.

En definitiva, s'ha obtingut un sistema que permet està sincronitzat al temps universal GPS cada segon i que, a més, permet actualitzar un cronòmetre, que corre també sobre temps GPS, cada 200ms.

6.-IMPLEMENTACIÓ

En aquest punt s'explicaran els diversos detalls d'implementació del projecte. Per una banda s'exposaran les diferents tecnologies utilitzades per desenvolupar cadascuna de les parts d'Iris així com altres alternatives que s'havien considerat.

Una vegada presentades les opcions, s'explicarà l'elecció realitzada i l'experiència viscuda amb la utilització d'aquestes tecnologies.

6.1.- Implementació aplicació pantalla

Com s'explica en aquesta memòria, l'aplicació de la pantalla hauria de ser força vistosa i fàcilment entenedora. És per això que es volia buscar algun llenguatge de programació que oferís les següents característiques:

- Havia d'oferir tècniques per dona sensació de dinamisme a l'aplicació.
- Donar facilitats per a la gestió de la informació entre les capes de vistes i domini.
- Oferir facilitats per a col·locar els elements a la pantalla, i per a moure'ls d'un lloc a l'altre, així com per a modificar la informació que s'hi mostra.

Amb aquestes condicions és obvi que el que necessitem és un llenguatge de programació orientat a l'elaboració de contingut gràfic. A continuació s'expliquen les diferents opcions considerades.

6.1.1.- Opcions considerades

Es van considerar tres opcions que complien amb els requisits anteriors, Flash, JavaFX i Silverlight, totes tres formen part del que s'anomenen RIA (Rich Internet Applications). Tot seguit s'expliquen les principals característiques de cadascuna.

6.1.1.1.- Flash

És un llenguatge multi plataforma suportat per la gran majoria de navegadors actuals. Està consolidat al mercat amb bastants anys d'experiència. Les dues principals avantatges que això comporta són, per una banda, que disposa de moltes fonts d'informació on buscar i documentar-se i, per l'altra, que serà un llenguatge que oferirà robustesa ja que la majoria de *bugs* que s'hagin pogut descobrir durant els anys estaran solucionats.

Per altra banda, està basat en ActionScript, un llenguatge una mica desorganitzat i bastant complex d'utilitzar. A més, té un baix rendiment gràfic. Finalment, un altre desavantatge és el cost del producte, ja que s'ofereixen alguns serveis gratuïts, però el producte sencer s'ha de pagar.

6.1.1.2.- JavaFX

Aquest és un producte de Sun Microsystems, gratuït i amb una sintaxi, com era d'esperar, molt semblant a la del propi Java. La principal avantatja que presenta és un servei anomenat *"binding"* que permet, entre d'altres coses, lligar dues variables, de manera que si una modifica el seu valor, automàticament l'altra també. Això ens seria de gran ajuda per a lligar les variables de vistes amb les de domini i així tenir automatitzat aquest mecanisme.

Per contra té l'inconvenient de ser un producte molt nou i, com a tal, existeix poca documentació a la web i segurament no haurà tingut temps de consolidar-se i solucionar els *bugs* que pugui tenir. Un altre inconvenient que presenta és que només pot funcionar sobre Windows o Mac, no corre sobre Linux.

6.1.1.3.- Silverlight

Està basat en Windows Presentation Foundation (WPF) i .Net FX 3.0. No té el problema de rendiment que presentava Flash ja que disposa d'acceleració per hardware a través de l'ús de DirectX[15]. A més, compta amb entorns de desenvolupament potents com ara VisualStudio.

Com a desavantatge trobem que només els navegadors més comuns Firefox, Safari I Internet Explorer el suporten. Un altre inconvenient que presenta és el seu cost ja que, com passava amb Flash, s'ha de pagar per adquirir el producte sencer.

6.1.2.- Alternativa triada i detalls d'implementació

Finalment , tot i ser un producte molt nou i amb poca experiència al mercat, s'ha decidit escollir JavaFX. Aquesta elecció ve donada per les següents raons:

- Primerament és un producte gratuït i que no fa falta adquirir cap tipus de llicència per utilitzar-lo.

- En segon lloc, la gran semblança amb Java, llenguatge que ja dominava qui havia d'implementar aquesta part.

- Finalment, i potser el motiu més important de tots, aquesta funcionalitat que ofereix en quant a lligar variables. L'equip ha cregut molt convenient per a l'aplicació comptar amb aquest servei.

Semblava que s'havia trobat el producte perfecte per al desenvolupament d'aquesta aplicació. Tot i això, a l'hora de la veritat van aparèixer certs contratemps que van fer endarrerir bastant la data d'acabament d'aquesta part.

Per una banda, una de les desavantatges que s'esmentaven quan es parlava de JavaFX, la falta d'informació a Internet. Quan hi havia un contratemps i es volia trobar una solució, aquesta moltes vegades no apareixia i s'havien de realitzar proves per entendre què estava passant. D'aquesta manera es perdia bastant de temps que feia que el projecte s'anés endarrerint.

En segon lloc, teníem el problema que l'aplicació funcionava però com més estona es trobava en funcionament més es ralentitzava la imatge fins que arribava un moment que es produïa un error del Heap de Java i deixava de funcionar l'aplicació.

L'equip va consumir una gran quantitat de temps per entendre d'on venia aquest problema i, finalment, es va descobrir que curiosament procedia de la seva principal avantatja, els *"bindings"*. El que passava era que quan s'esborraven elements de les vistes, si aquests tenien algunes variables lligades, el *Garbage Collector*[16] no les detectava com a que les havia d'esborrar. D'aquesta manera s'anava acumulant molta brossa a la memòria que acabava generant l'error de Heap que parlàvem anteriorment.

6.2.- Implementació aplicació PDAs

6.2.1.- Opcions considerades

A l'hora d'escollir la tecnologia que s'usaria per a desenvolupar la part de les PDAs, es van considerar dues alternatives, *Java MicroEdition* (JME) i C#. Tot seguit s'expliquen les principals característiques de cadascuna.

6.2.1.1.- Java MicroEdition

És un subconjunt del llenguatge Java amb un agregat de funcionalitats pròpies. Està orientat a l'elaboració d'aplicacions per a dispositius amb capacitats limitades de memòria i processat, és a dir mòbils, PDAs, etc. S'executa sobre una màquina virtual especial anomenada *Kilo Virtual Machine* (KVM), que és la màquina virtual més petita desenvolupada per Sun. El fet d'executar-se sobre una màquina virtual li dóna molta portabilitat. Per contra, no suporta punt flotant.

6.2.1.2.- C#

És un llenguatge orientat a objectes i d'última generació. A més és autocontingut, és a dir, no es necessiten referències externes d'altres llenguatges per a utilitzar-lo, ja que elimina elements que utilitzen altres llenguatges i que sense ells es redueix la complexitat a l'hora de programar.

Per altra banda, disposa d'un recol·lector de brossa com el conegut *Garbage Collector* de Java, per tant és innecessari disposar d'instruccions de destrucció d'objectes. Aquesta característica ofereix una certa tranquil·litat al programador, que no s'ha de preocupar per aquests temes.

6.2.2.- Alternativa triada i detalls d'implementació

Pel que fa a l'alternativa escollida, en aquesta part no s'ha pogut triar molt ja que les PDAs, que corren amb Windows Mobile, només acceptaven la segona opció per a ser programades. Així que, tot i que és bastant probable que s'hagués acabat utilitzant la tecnologia JME, per requeriments específics de les PDAs es va haver d'utilitzar C#.

A l'hora de la veritat, mentre s'estava implementant, no hi va haver masses problemes fora dels problemes comuns quan s'és inexpert en algun llenguatge. Els únics problemes van ser intentar des d'un principi utilitzar *sockets* per comunicar les PDAs, primer entre elles i posteriorment entre PDA i servidor. Però es va veure que aquest sistema no era possible ja que retardava molt l'intercanvi de dades. La solució va venir de la mà dels WebServices, que es el sistema que s'utilitza actualment i que ofereix bons resultats i amb certa rapidesa.

6.3.- Implementació servidor

Finalment, anem a parlar sobre la implementació del servidor. Abans de res però, explicar que el servidor del qual disposa WorldSensing és un Apache Tomcat. Això obliga a que l'aplicació que hi ha d'anar penjada estigui escrita en Java, per tant, en aquesta part no hi havia molt a elegir en quant al llenguatge utilitzat.

Per altra banda, s'ha utilitzat una biblioteca de Google anomenada Gdata API. Aquesta API proporciona serveis per a interactuar amb tota mena de dades com poden ser documents de text, fulls de càlcul, presentacions, inclús vídeos, etc. En el nostre cas ens centrarem però en els fulls de càlcul o, millor dit, els Spreadsheets de Google.

Alguns dels serveis que ofereix la API Gdata en quant a Spreadsheets es refereix són:

- Connexió a un Spreadsheet concret mitjançant el nom d'usuari i la contrasenya del creador o qui sigui que pugui accedir a aquest Spreadsheet.

- Navegar per les pàgines d'un Spreadsheet o accedir a una de concreta cercant-la a través del seu nom.

- Realitzar consultes dintre d'una fulla per files i/o per columnes i extreure'n la informació del contingut de les celes afectades.

- Insertar nova informació a una cela o conjunt de celes.

Per últim es necessitava alguna tecnologia per a insertar codi dinàmic a una pàgina web estàtica. Això és necessari per implementar l'aplicació web de visualització d'històrics de competicions, on tenim un esquelet de la pàgina web (part estàtica) que falta ser omplert amb les competicions que s'hagin de mostrar en un moment donat (part dinàmica).

Hi ha diverses maneres d'implementar això les principals i unes de les més conegudes en són 2:

- ***JavaServlets***: Classes Java que inserten codi html a una web.

- ***Java Server Pages (JSPs)***: Mecanismes per insertar codi Java a una pàgina html. Aquest codi s'executarà a la part del servidor.

De les dues opcions és obvi que és molt més simple treballar amb la segona, ja que resulta molt més simple insertar codi Java a dintre un html que anar realitzant println's amb codi html a l'interior. Per tant s'ha elegit treballar amb JSP per a aquesta part.

6.3.1.- Detalls d'implementació

Aquesta part és una de les que menys problemes ha donat al moment d'implementar-la. Tot i això sí que hi ha hagut algun contratemps.

Per una banda es va donar el problema que les crides a Spreadsheets amb gran volum de dades es ralentitzaven en gran mesura. En aquest moment es va realitzar un estudi de la màxima quantitat de dades amb la que es podria trobar l'aplicació. Un cop es tenia aquest valor es va realitzar una prova representativa i es va veure que el sistema podria funcionar amb total normalitat.

Tot i això va ser en aquest moment quan es va pensar en posar una Base de Dades sobre la qual realitzar les peticions de consulta de dades. Com que es va veure que era una millora clara per al sistema es va decidir tirar-ho endavant.

Per altra banda, un altre problema que presenta la Gdata API és la impossibilitat de crear noves files o columnes a un Spreadsheet a través de codi. Aquesta mancança va provocar un canvi al sistema i es que es volia poder insertar noves participacions (i per tant noves files) des de les PDAs.

Finalment això no serà possible, en canvi si que es podrà realitzar interactuant directament amb el Spreadsheet.

7.-PROVES

7.1.- Objectiu de les proves

L'objectiu de les proves que es realitzaran és comprovar el correcte funcionament del sistema. Això inclou des del correcte funcionament de cada aplicació per separat fins al correcte funcionament de tot el sistema integrat.

A part, es vol provar el sistema en varietat d'escenaris. És per això que es construiran diferents competicions amb particularitats per a cadascuna per poder observar la resposta d'Iris envers aquestes.

7.2.- Metodologia de les proves

La metodologia de les proves realitzades consisteix en anar augmentant el grau d'integració del sistema a mesura que es comprova que el grau d'integració anterior ja s'ha estabilitzat. A més, com marca un dels objectiu, el sistema serà provat amb un joc de proves consistent en un conjunt de competicions diferents entre sí.

Primerament es realitzaran proves sobre les diverses parts del sistema per separat, és a dir, per una banda les PDAs, el servidor i finalment la pantalla.

Una vegada s'hagi estabilitzat aquesta part, és a dir, funcioni correctament cada element per la seva banda, s'incrementarà el grau d'integració del sistema provant el funcionament de les PDAs amb el servidor i, per altra banda, de la pantalla amb el servidor.

Assolit aquest grau es realitzarà una prova final del sistema amb tots els elements funcionant al mateix temps. En quan s'hagi aconseguit l'èxit serà el moment de provar el sistema en una situació real. És a dir assistir a una competició que es realitzi i posar en funcionament el sistema.

8.- PRODUCTE FINAL

En aquests moments el projecte es troba en fase final de proves on s'estan polint tots aquells errors que s'han anat trobant. Tot i això el producte està pràcticament acabat i a continuació es mostraran captures de pantalla de les diverses aplicacions que constitueixen Iris.

8.1.- Aplicació PDAs

Aquestes són les captures de pantalla de les vistes pertanyents al software de les PDAs.

Imatge 8.1: Vista selecció competició

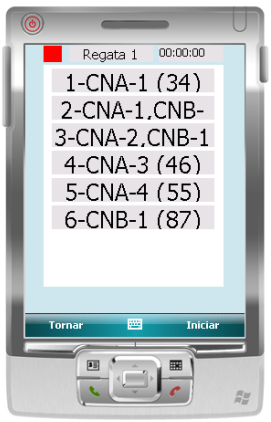
La primera vista, la que es veu a la figura 8.1, mostra les diferents competicions que existeixen al sistema i es on s'ha de seleccionar la que es vol utilitzar.

Imatge 8.2: Vista selecció rol

Posteriorment passem a la pantalla de selecció de rol, la que mostra la imatge 8.2, on informem al sistema de quin rol anem volem ser. Segons la selecció que haguem fet, les properes vistes serà unes o unes altres.

Imatge 8.3: Vista selecció regata

La imatge 8.3 mostra la vista de selecció de regata. Aquesta vista conté les regates pertanyents a la competició que hem seleccionat anteriorment. A més mostra l'estat de cada regata, si està acabada, pendent o en curs.



Imatge 8.4: Vista iniciar regata race

Imatge 8.5: Vista iniciar regata rally

Les imatges 8.4 i 8.5 mostren les diferents vistes d'inici de regata que es visualitzen depenent de si estem tractant amb una regata tipus *race* o tipus *rally.* Per una banda, la imatge 8.4 és la vista per iniciar regates tipus *race* i l'única acció que es pot dur a terme en aquesta vista és clicar al botó *Iniciar*, que iniciarà la regata. Per altra banda tenim la vista de la imatge 8.5 que inicia regates tipus *rally*, en la qual primerament s'ha de seleccionar el participant i posteriorment prémer el botó *Iniciar* que iniciarà la regata només per al participant seleccionat.

Imatge 8.6: Vista arribada regata

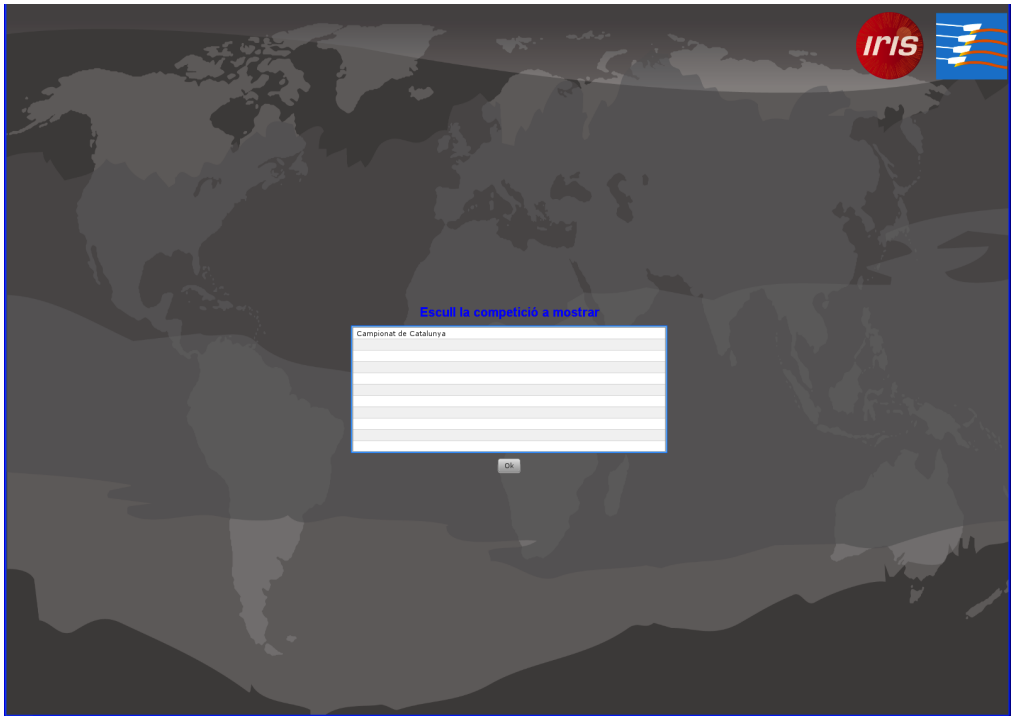
Finalment tenim la vista que mostra la imatge 8.6, on es realitzen les arribades. Aquesta vista és la mateixa per a arribades de regates de tipus *race* o *rally*. El que mostra la vista és la llista d'arribades fins al moment i el crono des de l'inici de la regata. La manera de procedir és la següent: Es clica el botó *Arribada!* que crearà una nova arribada a la pantalla amb un *?* com a participant. Clicant a sobre aquest interrogant apareixerà la llista de participants que encara no han arribat i se'n seleccionarà un.

Cal dir també que s'ha previst que es puguin produir dues arribades molt ajustades amb molt poc temps entre elles. Llavors, com que el procediment anterior costa el seu temps, es permeten realitzar vàries arribades sense identificar de qui han estat aquestes. Posteriorment és responsabilitat dels àrbitres recordar l'ordre d'arriada i assignar cada crono al participant corresponent.

En definitiva, com es pot observar es tracta de vistes simples i intuïtives que no suposaran pràcticament cap esforç d'aprenentatge per part dels àrbitres de la federació. I a més permetran que els temps capturats siguin molt més precisos que actualment.

8.2.- Aplicació pantalla

Com a producte final de la part de presentació d'informació al públic tenim aquesta aplicació que té les característiques que es demanaven a priori. Es vistosa, bastant intuïtiva i dona la sensació de dinamisme que es buscava. Tot seguit es mostra i es detalla el funcionament d'aquesta aplicació.



Imatge 8.7: Vista selecció competició

La imatge 8.7 mostra la vista de selecció de competicions, on s'escull la competició desitjada i es prem *Ok* per passar a la finestra de visualització de les regates. Una vegada fet això la pantalla començarà a mostrar l'estat de la competició amb el format que es mostra a continuació.

Imatge 8.8: Vista estat competició

I aquesta és la vista que mostra l'estat de les regates. Es distingeixen 5 parts diferents que s'expliquen tot seguit:

- ***Títol competició:*** És on es mostra el títol que se li ha introduït a la competició des del Spreadsheet de Google Docs.

- ***Zona regates futures:*** En aquesta zona es mostren les regates que estan pendents de realitzar-se en un breu període de temps. Estan ordenades de dalt a baix de més llunyana a més propera, de manera que la regata que més a prop està de la zona de regates en curs serà la següent en realitzar-se.

- ***Zona regates en curs:*** Aquí és on es mostren les regates que s'estan realitzant en un moment donat. Cadascuna d'aquestes té o bé un crono de la regata si aquesta és tipus *race* o bé un crono per a cada participant en cas que la regata sigui tipus *rally*.

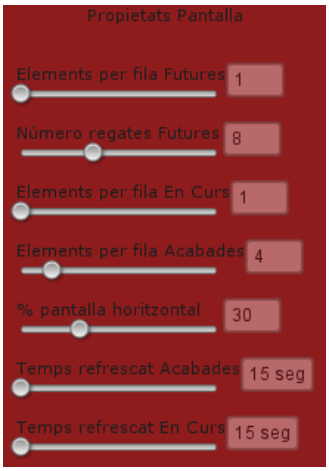
- ***Zona regates finalitzades:*** Les regates finalitzades es mostren en aquesta zona on es van acumulant després d'haver-se acabat. Aquí els participants estaran ordenats per la posició que hagin obtingut a la regata.

  Cal dir a més que tant la zona de regates en curs com la de regates finalitzades, si es

dona el cas que totes les regates que s'han de mostrar no caben a la pantalla, automàticament s'engega un *timer* que va refrescant la part afectada realitzant un bucle amb totes les regates a mostrar.

- ***Panell de control:*** Aquesta zona, que quan es clica amb el ratolí mostra una vista com la de la imatge 8.9, presenta les següents propietats de la pantalla que es poden modificar.

  - **Elements per fila Futures**: Número de columnes que es mostraran a la zona de regates futures (per defecte 1).
  - **Número regates Futures**: Número de regates que apareixeran a la zona de regates futures (per defecte 8).
  - **Elements per fila En Curs**: Número de columnes que es mostraran a la zona de regates en curs (per defecte 1).
  - **Elements per fila Acabades**: Número de columnes que es mostraran a la zona de regates futures (per defecte 4).
  - **% pantalla horitzontal**: Percentatge dedicat a la zona esquerra (zona regates futures + zona regates en curs). El percentatge dedicat a la zona de regates finalitzades és el que queda fins arribar al 100% (per defecte 30% esquerra i 70% dreta).
  - **Temps refrescat acabades**: Cada quan es refresca la part de regates acabades en cas que necessiti aquesta opció (per defecte 15 segons).
  - **Temps refrescat en curs**: Cada quan es refresca la part de regates en curs en cas que necessiti aquesta opció (per defecte 15 segons).



Imatge 8.9: Panell de control

8.3.- Aplicació web

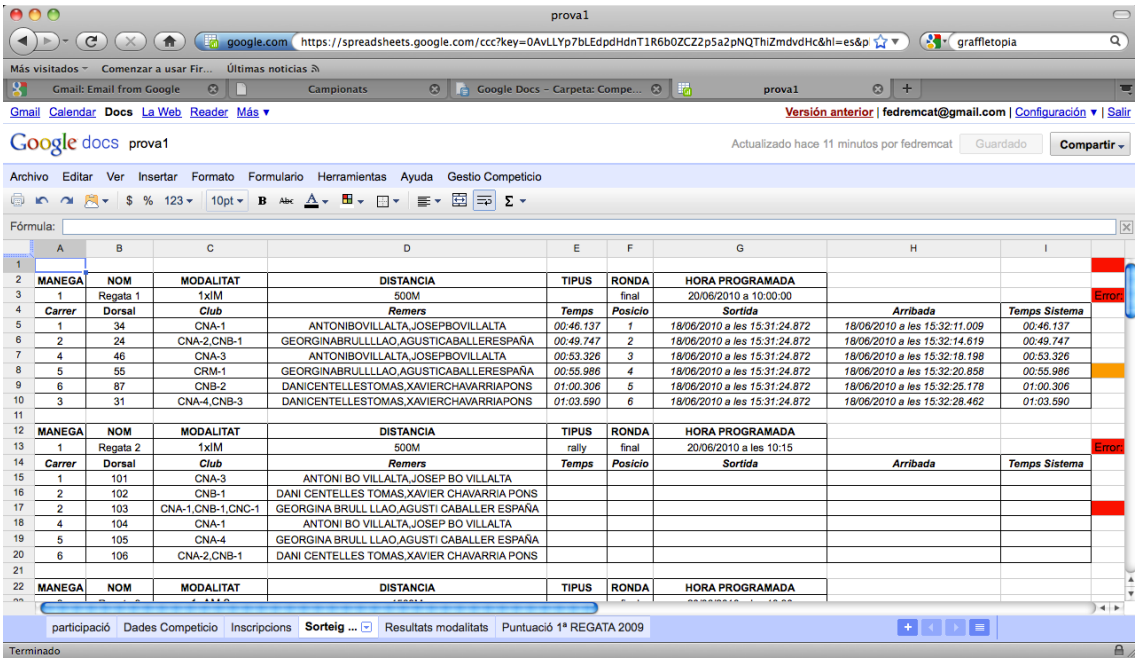
No menys important és la web que mostra l'històric de competicions. Aquesta, que com es veurà segueix el mateix patró gràfic que l'aplicació de la pantalla, també compleix amb els requeriments que s'especificaven en el seu moment. Tot i això, la part de generació d'estadístiques no s'ha realitzat i es podria tenir en compte per a una segona versió del producte.

Imatge 8.10: Web d'històric de competicions

Actualment aquesta aplicació web permet veure les dades d'una competició (nom, data, organitzadors,...) així com el llistat de totes els regates classificades per modalitat i ordenades per data de realització.

8.4.- Aplicació GoogleDocs

Aquí tenim unes captures mostrant com es representa finalment la informació de les competicions als Spreadsheets de GoogleDocs i de quines accions es poden realitzar d'es d'aquesta aplicació.



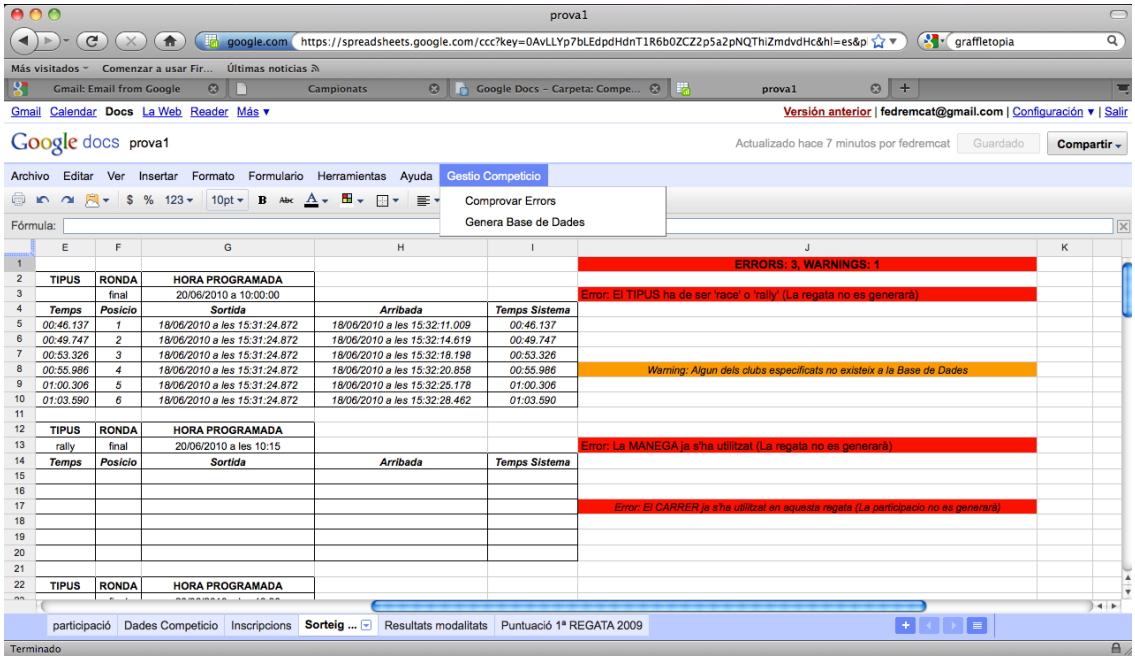
MANEGA	NOM	MODALITAT	DISTANCIA	TIPUS	RONDA	HORA PROGRAMADA		
1	Regata 1	1xIM	500M		final	20/06/2010 a 10:00:00		Error:
Carrer	*Dorsal*	*Club*	*Remers*	*Temps*	*Posicio*	*Sortida*	*Arribada*	*Temps Sistema*
1	34	CNA-1	ANTONIBOVILLALTA,JOSEPBOVILLALTA	*00:46.137*	*1*	*18/06/2010 a les 15:31:24.872*	*18/06/2010 a les 15:32:11.009*	*00:46.137*
2	24	CNA-2,CNB-1	GEORGINABRULLLLAO,AGUSTICABALLERESPAÑA	*00:49.747*	*2*	*18/06/2010 a les 15:31:24.872*	*18/06/2010 a les 15:32:14.619*	*00:49.747*
4	46	CNA-3	ANTONIBOVILLALTA,JOSEPBOVILLALTA	*00:53.326*	*3*	*18/06/2010 a les 15:31:24.872*	*18/06/2010 a les 15:32:18.198*	*00:53.326*
5	55	CRM-1	GEORGINABRULLLLAO,AGUSTICABALLERESPAÑA	*00:55.986*	*4*	*18/06/2010 a les 15:31:24.872*	*18/06/2010 a les 15:32:20.858*	*00:55.986*
6	87	CNB-2	DANICENTELLESTOMAS,XAVIERCHAVARRIAPONS	*01:00.306*	*5*	*18/06/2010 a les 15:31:24.872*	*18/06/2010 a les 15:32:25.178*	*01:00.306*
3	31	CNA-4,CNB-3	DANICENTELLESTOMAS,XAVIERCHAVARRIAPONS	*01:03.590*	*6*	*18/06/2010 a les 15:31:24.872*	*18/06/2010 a les 15:32:28.462*	*01:03.590*

MANEGA	NOM	MODALITAT	DISTANCIA	TIPUS	RONDA	HORA PROGRAMADA		
1	Regata 2	1xIM	500M	rally	final	20/06/2010 a les 10:15		Error:
Carrer	*Dorsal*	*Club*	*Remers*	*Temps*	*Posicio*	*Sortida*	*Arribada*	*Temps Sistema*
1	101	CNA-3	ANTONI BO VILLALTA,JOSEP BO VILLALTA					
2	102	CNB-1	DANI CENTELLES TOMAS,XAVIER CHAVARRIA PONS					
2	103	CNA-1,CNB-1,CNC-1	GEORGINA BRULL LLAO,AGUSTI CABALLER ESPAÑA					
4	104	CNA-1	ANTONI BO VILLALTA,JOSEP BO VILLALTA					
5	105	CNA-4	GEORGINA BRULL LLAO,AGUSTI CABALLER ESPAÑA					
6	106	CNA-2,CNB-1	DANI CENTELLES TOMAS,XAVIER CHAVARRIA PONS					

MANEGA	NOM	MODALITAT	DISTANCIA	TIPUS	RONDA	HORA PROGRAMADA

Imatge 8.11: Representació de les regates

La imatge 8.11 mostra com es representen les regates al full de càlcul. Com es va demanar a la presa de requeriments, aquesta representació és bastant semblant a la que utilitzaven anteriorment la FCR. Tot i això ara, quan es comet algun error a l'hora d'inserir els camps l'aplicació ho detecta. A més a més, com es mostrarà a continuació, avisa l'usuari de l'error que s'ha donat.



Imatge 8.12: Visualització d'errors i menú d'opcions

La imatge 8.12 ensenya alguns exemples d'aquests errors que es poden cometre i que el sistema reconeix i comunica a l'usuari. A més també mostra una cela a la primera línia on s'informa de diferents aspectes com ara: Número d'errors i warnings trobats, si la base de dades ha estat generada amb èxit o no,...

Per altra banda, es pot observar que disposa d'una pestanya anomenada Gestió Competicions des d'on es poden realitzar dues accions:

- ***Comprovar errors:*** Quan es clica aquesta opció, el sistema parseja la competició en la que ens trobem i mostra a continuació de cada línia si aquesta conté algun error o warning i una breu explicació d'aquest.

- ***Generar Base de Dades:*** Aquesta crida realitzarà una còpia exacta a la base de dades del que està representat en un moment donat al full de càlcul.

9.- RELACIÓ AMB ASSIGNATURES DE LA CARRERA

Clarament les assignatures que més es veuen reflectides en aquest projecte són les que fan referència a Enginyeria del Software i Bases de Dades, però no són les úniques.

A part de les típiques BD, ES1 i ES2 també han anat bé els coneixements adquirits a *Disseny de sistemes basats en el web* (DSBW) on s'expliquen les diverses tècniques i maneres de desenvolupar una aplicació web., però no són les úniques.

Per altra banda, tot el coneixement adquirit a *Programació concurrent i distribuïda* (PCD) ha estat productiu també per a aquest projecte. Per exemple, per detectar als possibles problemes d'accés concurrent a les dades i dissenyar una solució basada en adquisició i alliberament del domini per a evitar-los.

Finalment també ha resultat productiva l'assignatura *Planificació i gestió de projectes i sistemes informàtics* (PGPSI) en la qual s'explicaven els passos a realitzar a l'hora d'abordar un projecte d'aquestes dimensions, és a dir, com passar de la idea al producte final. Cal dir que aquests passos s'han intentat seguir, en la mesura del possible, en el nostre projecte.

10.-CONCLUSIONS

Tots els objectius que es marcaven a l'inici d'aquesta memòria s'han aconseguit finalment, així que creiem que el producte tindrà èxit dintre la FCR i qui sap si en alguna federacions de rem més o inclús d'altres esports.

La implantació de les PDAs millorarà substancialment la precisió dels cronos de les competicions. A més la pantalla aportarà una gran informació a un públic que actualment havia d'estar pendent en tot moment del lloc on es realitzava la prova.

Al dissenyar un projecte d'aquesta grandària és important definir cada cosa al seu moment i preveient els problemes que hi podran aparèixer ja que un error de disseny pot aportar gran quantitat de canvis a l'hora de la implementació.

Pel que fa l'elecció de JavaFX, un llenguatge acabat de nàixer, per a implementar l'aplicació de la pantalla, ha quedat demostrat que, en cas de necessitar una aplicació seriosa, és millor utilitzar alguna tecnologia que estigui consolidada al mercat, és a dir, que sigui una tecnologia més madura.

Per altra banda, Iris combina una gran varietat de tecnologies diferents. Haver d'aprendre el funcionament de cadascuna m'ha aportat per una banda coneixement, expandint el ja adquirit durant la carrera. Per altra banda, predisposició a realitzar el que s'anomena autoaprenentatge, un terme molt utilitzat dintre l'àmbit de la informàtica.

11.- GLOSSARI

[1]Regates: Cadascuna de les proves en les què es divideix una competició de rem.

[2]PDA: De l'anglès *Personal Digital Assistant* (Assistent Digital Personal), és una computadora de mà creada originalment com agenda electrònica. D'entre les diferents característiques les més importants són:

- Permet comunicar-se amb ella directament a través de la pantalla.
- Pot funcionar com a mòbil ja que accepta targetes SIM.
- Funciona amb un sistema operatiu tipus Windows mobile.

[3]GPS: De l'anglès Global Positioning System (Sistema de posicionament global), és un sistema de navegació per satèl·lit que permet determinar la posició d'un objecte a qualsevol part del món. A més té atribuït un rellotge universal de referència a tot el món. Aquesta part del GPS, el rellotge, és el que s'utilitza en aquest projecte.

[4]WebServices: És un conjunt d'estàndards i protocols que serveixen per intercanviar dades entre aplicacions les quals, normalment, estan a diferent lloc físic i es comuniquen a través d'Internet. Està dissenyat de tal manera que permet l'intercanvi de dades entre aplicacions que estan escrites en diferents llenguatges de programació.

[5]Checkpoint: Significa punt de control i al context d'aquest projecte és cadascun dels punts on es controla el temps de pas dels participants d'una regata.

[6]Split: Fa referència a l'acció de capturar el temps, ja sigui d'inici o de fi, d'un participant.

[7]Parser: Part d'una aplicació software encarregada de rebre les dades codificades d'alguna manera i extreure'n la informació desitjada.

[8]Spreadsheets: Són els típics fulls de càlcul coneguts principalment per l'aplicació Excel de Microsoft Office. Mostra múltiples celes en les quals es pot posar informació i realitzar operacions sobre aquesta com ara agregats, dibuixar gràfiques i moltes altres funcionalitats.

[9]GoogleDocs: És una aplicació basada en web creada, com el seu nom indica, per Google. Permet gestionar documents en línia, però la funcionalitat que més el destaca és la possibilitat que ofereix de modificar informació en grup i al mateix temps, de manera que el que un modifica es fa visible per als altres instantàniament. Per altra banda, també ofereix serveis per a poder ser gestionat a través de codi.

[10]API: De l'anglès Application Programming Interface (Interfície de programació d'aplicacions), és un conjunt de funcionalitats que ofereix alguna biblioteca per ser utilitzada per un altre software com a capa d'abstracció.

[11]GPRS: De l'anglès General Packet Radio Service (Servei General de Paquets via Radio), és un sistema per a la transmissió de dades per paquets dissenyat per a aparells mòbils. Permet velocitats de transferència de 56 a 144 kbps.

[12]Templates: Patró, peça base destinat a la creació de documents d'unes característiques concretes.

[13]Sockets: Tipus de comunicació entre dos aplicacions que, per norma general, estan en diferents màquines. Un socket queda definit per una adreça IP, un protocol de transport i un número de port.

[14]Model-vista-controlador: Patró de disseny que separa el que són les dades (model), la interfície d'usuari (vista) i la lògica de control (controlador). Aquest patró és molt utilitzat en aplicacions web. A continuació es mostra un diagrama UML explicatiu d'aquest patró.

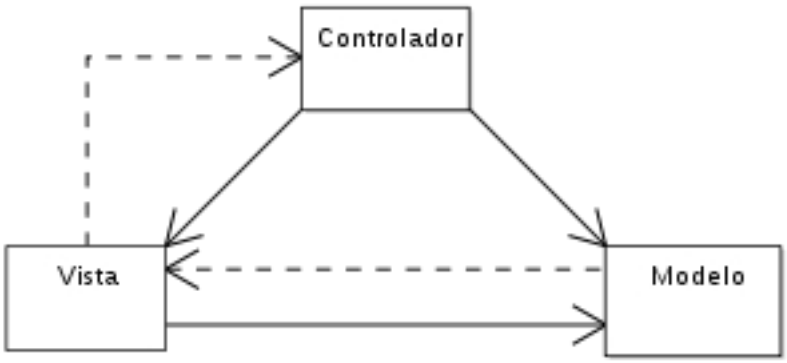


Diagrama 11.1: UML patró model-vista-controlador

[15]DirectX: És una col·lecció d'API creades per facilitar les complexes tasques relacionades amb la multimèdia, especialment pel que fa a la part gràfica.

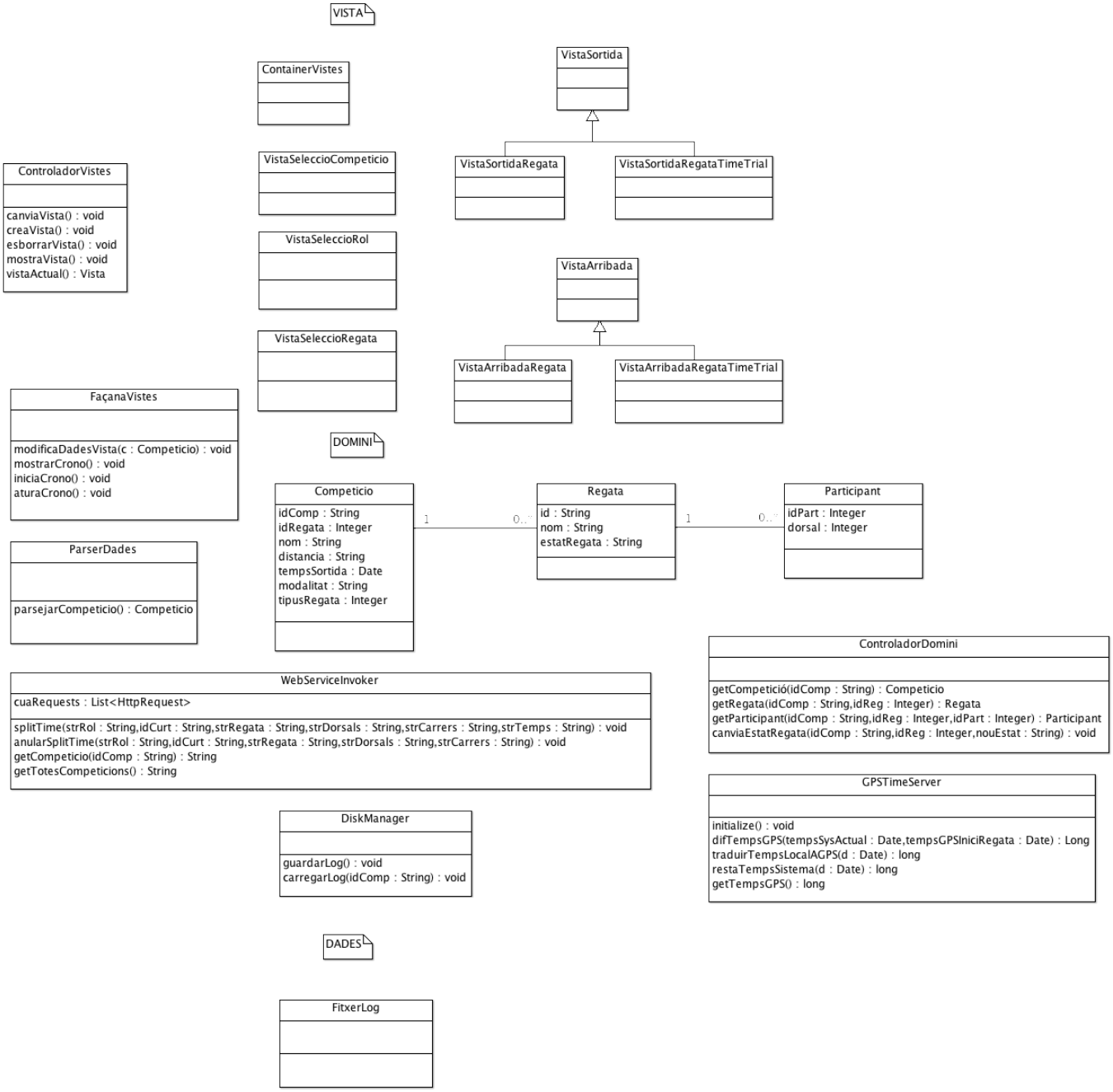
[16]Garbage Collector: Significa recol·lector de brossa i és un mecanisme de gestió de memòria implementat per alguns llenguatges de programació de tipus interpretat com ara Java. Facilita la feina als programadors ja que aquests no s'han de preocupar d'esborrar els objectes una vegada ja no s'hagin de fer servir més.

12.- BIBLIOGRAFIA

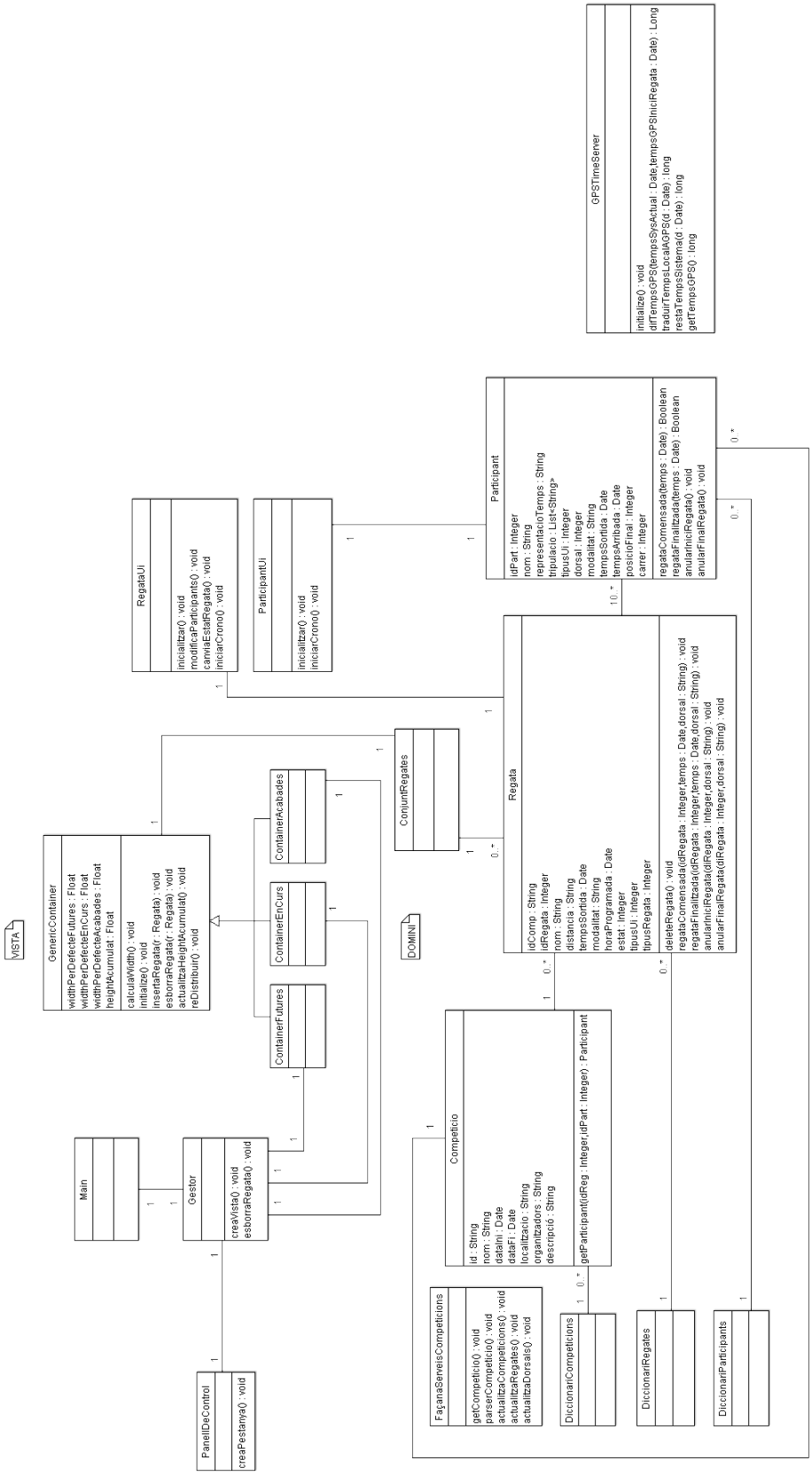
- http://code.google.com/intl/es-ES/apis/gdata/
- http://code.google.com/intl/es-ES/apis/spreadsheets/
- http://www.jsptut.com/
- http://javafx.com/
- http://www.csharp-station.com/Tutorial.aspx
- http://www.swisstiming.com/
- http://www.remcatalunya.org/web/
- http://tomcat.apache.org/
- http://es.wikipedia.org/wiki/Wikipedia:Portada
- Pro JavaFX™ Platform: Script, Desktop and Mobile RIA with Java™ Technology by James L. Weaver, Weiqi Gao, Stephen Chin, and Dean Iverson (edició – 21 Juliol, 2009).
- Design Patterns: Elements of Reusable Object-Oriented Software by Erich Gamma, Richard Helm, Ralph Johnson, and John M. Vlissides (edició – 10 Novembre , 1994).
- Design Patterns in Java(TM) (Software Patterns Series) by Steven John Metsker and William C. Wake (edició – 28 abril, 2006)

13.- ANNEXOS

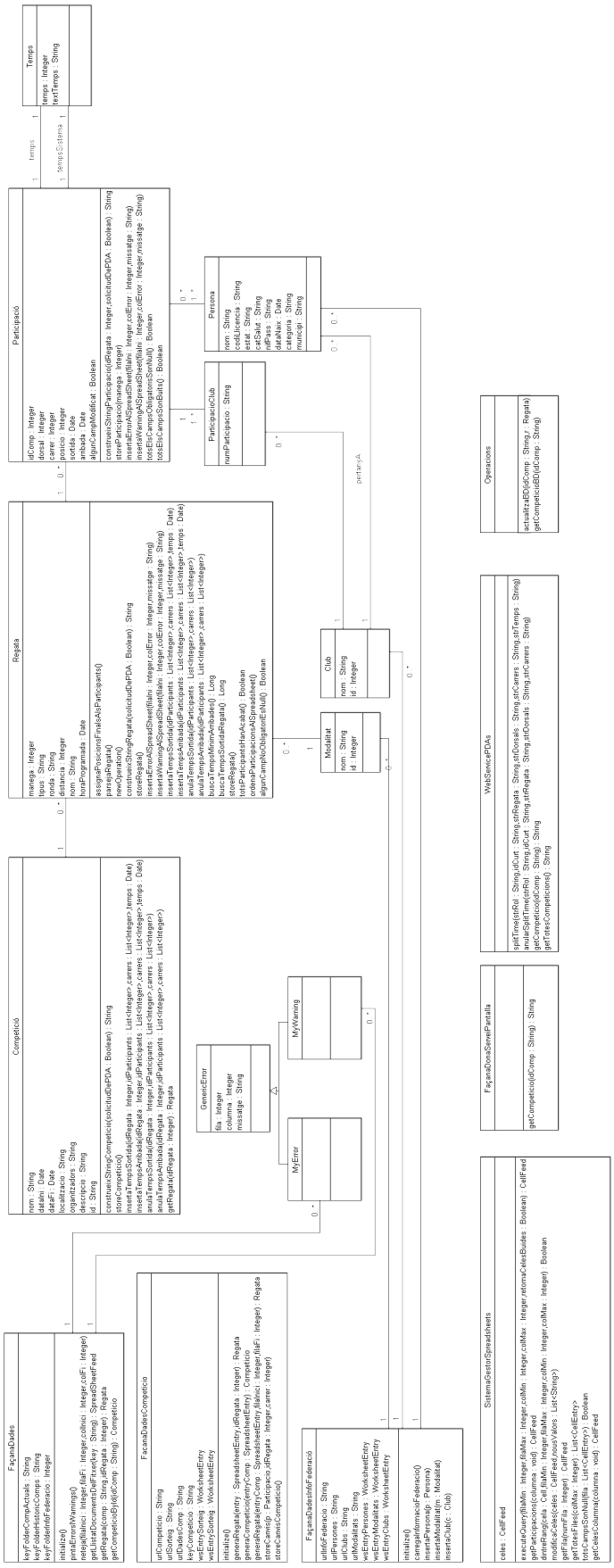
13.1.- UML PDAs



13.2.- UML Pantalla



13.3.- UML Dades



AGRAÏMENTS

El desenvolupament d'aquest projecte no hagués estat possible sense l'ajuda oferida en tot moment per part de dos membres de l'empresa WorldSensing. Per una banda, el meu director de projecte Xavier Vilajosana que en tot moment ha estat allí quan el necessitava. Per altre costat, una altra persona que ja ha estat nombrada en aquesta memòria, Jordi Llosa, que gràcies als seus coneixements sobre el rem i més encara informàtics ha estat un altre puntal de referència en cas de dubte.

Finalment agrair també el suport de tota la meva família que m'ha animat i encoratjat en tot moment, tant durant la realització d'aquest projecte com durant tota la carrera.